

# A Web Anonymizer Platform for Datasets with Personal Information

Christophe da Silva Ferreira

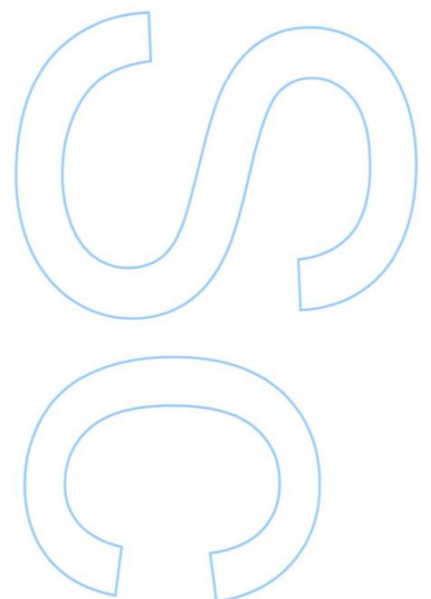
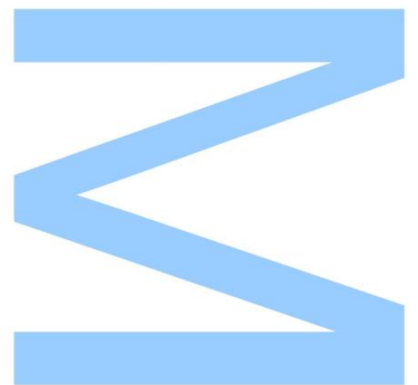
Mestrado Integrado em Engenharia de Redes e Sistemas Informáticos  
Departamento de Ciência de Computadores  
2017

## **Orientador**

Manuel Eduardo Carvalho Duarte Correia, Professor Auxiliar  
Faculdade de Ciências da Universidade do Porto

## **Coorientador**

Luís Filipe Coelho Antunes, Professor Associado  
Faculdade de Ciências da Universidade do Porto

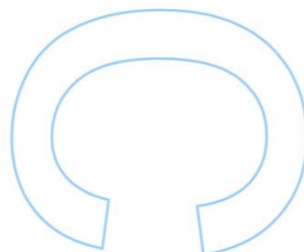
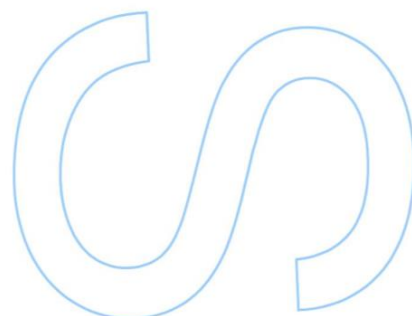
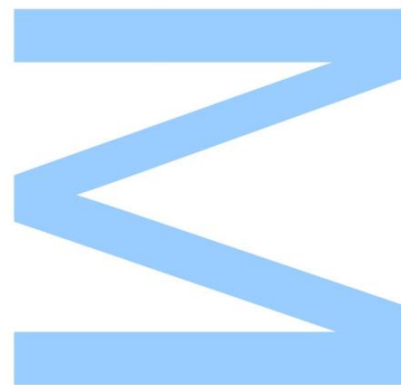




Todas as correções determinadas pelo júri, e só essas, foram efetuadas.

O Presidente do Júri,

Porto, \_\_\_\_/\_\_\_\_/\_\_\_\_





To Nini

# Acknowledgments

I would like to thank some persons that were essential to accomplish this dissertation, not only in this past year, but for a long time.

It is impossible to verbalize my gratitude to everything my mother sacrificed for me, I always could count on her, for that, all my love and appreciation to her. I want to thank also, my brother and sister for all their support, without your help this journey would be impossible.

I would like to express my sincere gratitude to the *Computer Science Department* exceptional group of professors, particularly to my supervisors, Prof. Manuel Correia and Prof. Luís Antunes. My deepest gratitude for all of their support, time and advice throughout this year.

Finally, I would like to thank all the persons that somehow influenced me during my life.

# Abstract

The migration of societal processes to the Internet, the massification of digital services and more recently, Internet of Things (IoT) devices in the form of personal sensors, has changed completely the way personal information is collected, stored and used. The exponential growth on the amount of personal data that is thus collected, opens new possibilities on the way it can be used for scientific research, or otherwise more mundane commercial purposes. However, special care must be taken because personal privacy is a basic human right that is strongly protected by Law. There is, therefore, a high demand for privacy aware solutions that allows for the safe and lawful re-use of *datasets* based on personal information. One can argue that one way to comply with the law resides in the appropriate application of *de-identification* techniques, as a way of guaranteeing privacy, by deriving useful *de-identified datasets* that still has enough information to be useful.

The goal of this dissertation is to describe the development of a *web anonymization* application, that simplifies the *de-identification* of *datasets* containing personal information. First, well-known available desktop solutions were analyzed, in order to choose the most adequate and complete, that could be used as a strong base for a web *de-identification* platform. We found that *ARX* is a desktop *de-identification* platform that fulfils our requirements. The *de-identified datasets* produced by *ARX* were then tested, in terms of resistance to well-known *re-identification* attacks, and the results thus obtained were deemed satisfactory. *ARX* also has an interface API that was integrated into a REST based API for our *Web Anonymizer* platform, to support a responsive web interface that mimics the interface found on the original *ARX* desktop application.

Finally, we performed a series of tests in order to verify if the web application produced results were similar to its desktop counterpart, the execution times were acceptable when compared to the original desktop application. We concluded the *Web Anonymizer* fulfils its initial objectives. However, as expected, the execution times for the platform created were longer than the desktop *ARX* times. This is solely due to the network and REST API induced delays, because

the library supporting the core *de-identification* algorithms remained the same. However, this increase does not compromise practicality, because execution times remain well within reasonable end-user usability constraints. Some *de-identification* configurations available on *ARX* were not implemented in this version of the *Web Anonymizer*. This caused a slight decrease in the *datasets re-identification* resistance when compared to the ones produced by the desktop *ARX*.

**Keywords:** De-identification, Anonymization, Pseudonymization, Dataset, Re-identification, Personal Information

# Resumo

A migração de processos sociais para a Internet, a massificação de serviços digitais e mais recentemente, os dispositivos IoT sob a forma de sensores pessoais, mudou completamente a maneira como as informações pessoais são recolhidas, armazenadas e usadas. O crescimento exponencial de dados pessoais recolhidos, abre assim novas possibilidades na forma em que estes podem ser utilizados para pesquisas científicas, ou outros fins comerciais mais mundanos. No entanto, é necessário ter um cuidado especial porque a privacidade é um direito humano básico que está fortemente protegido por lei. Existe, portanto, uma grande procura por soluções de privacidade que permitam a reutilização segura e legal de *datasets* que contenham informações pessoais. Podemos argumentar que uma maneira de cumprir a lei, reside na aplicação adequada de técnicas de *de-identification*, como forma de garantir a privacidade, obtendo *datasets* seguros que ainda possuem informações suficientes para sejam úteis.

O objetivo desta dissertação é descrever o desenvolvimento de uma aplicação *web* para anonimização de dados, que simplifica a *de-identification* de *datasets* contendo informações pessoais. Primeiro, foram analisadas as soluções desktop disponíveis mais conhecidas, para escolher a mais adequada e completa, de modo a ser utilizada como uma base sólida para uma plataforma *web* de *de-identification*. Descobrimos que o *ARX* é uma plataforma desktop de *de-identification* que satisfazia aos nossos requisitos. Os *datasets* produzidos pelo *ARX* foram então testados, em termos de resistência a ataques de *re-identification* bem sucedidos, os resultados assim obtidos foram considerados satisfatórios. O *ARX* também possui uma API, que foi integrada numa REST API, servindo de base para a plataforma *Web Anonymizer*, garantindo uma interface *web* responsiva que imita a interface encontrada na aplicação de desktop original.

Finalmente, realizamos uma série de testes para verificar se a aplicação *web* produzia resultados semelhantes à versão desktop, os tempos de execução foram aceitáveis em comparação com a aplicação original. Concluímos que o *Web Anonymizer* cumpre os seus objetivos iniciais. No entanto, como esperado, os tempos de execução da plataforma criada foram maiores que os



tempos do *ARX*. Isso é devido aos atrasos introduzidos pela rede e a API REST, isto porque a biblioteca que suporta os principais algoritmos de *de-identification* permaneceu igual. No entanto, esse aumento não compromete a praticidade, porque os tempos de execução permanecem bem dentro das restrições razoáveis de usabilidade do usuário final. Algumas configurações de *de-identification* disponíveis no *ARX* não foram implementadas nesta versão do *Web Anonymizer*. Isso causou uma ligeira diminuição na resistência dos *datasets* à *re-identification* quando comparada com os produzidos pela versão desktop do *ARX*.

"If after I die, people want to write my biography, there is nothing simpler. They only need two dates: the date of my birth and the date of my death. Between one and another, every day is mine."

Fernando Pessoa



# Contents

<b>Acknowledgments</b>	<b>V</b>
<b>Abstract</b>	<b>VI</b>
<b>Resumo</b>	<b>VIII</b>
<b>Table of Contents</b>	<b>XII</b>
<b>List of Figures</b>	<b>XV</b>
<b>List of Tables</b>	<b>XVI</b>
<b>Acronyms</b>	<b>XVII</b>
<b>Introduction</b>	<b>2</b>
1.1. Context . . . . .	2
1.2. Motivation . . . . .	4
1.3. Objectives . . . . .	5
<b>Background</b>	<b>7</b>
2.1. <i>De-identification</i> . . . . .	7
2.1.1. <i>Anonymization</i> . . . . .	8
2.1.2. <i>Pseudonymization</i> . . . . .	9
2.2. <i>Re-identification</i> . . . . .	9
2.3. Data Protection Officer . . . . .	10
2.4. Legislation . . . . .	11
2.4.1. Worldwide Data Privacy Laws . . . . .	12
2.4.2. EU vs. USA . . . . .	14

2.4.2.1. European Data Protection Regulation . . . . .	18
2.4.3. Portuguese Legislation . . . . .	19
2.5. Personal Information Breaches . . . . .	20
2.5.1. <i>Netflix</i> . . . . .	20
2.5.2. <i>TRICARE</i> . . . . .	21
<b>State of the Art</b>	<b>23</b>
3.1. Privacy Models . . . . .	23
3.2. Hierarchies . . . . .	25
3.3. <i>De-identification</i> Solutions . . . . .	26
3.3.1. Open Source . . . . .	26
3.3.2. Private Solutions . . . . .	33
3.3.3. Tools Comparison . . . . .	34
3.3.4. Final Results . . . . .	38
<b>Development</b>	<b>40</b>
4.1. <i>Web Anonymizer</i> . . . . .	40
4.1.1. Architecture . . . . .	40
4.1.2. Interface . . . . .	42
4.1.3. Technologies . . . . .	47
4.1.3.1. <i>Client-side</i> . . . . .	47
4.1.3.2. <i>Server-side</i> . . . . .	50
4.1.4. <i>ARX API</i> . . . . .	54
4.1.5. <i>Web Anonymizer Overview</i> . . . . .	56
4.1.5.1. Adapting the <i>ARX API</i> . . . . .	57
4.1.5.2. <i>RESTful API</i> . . . . .	62
4.1.6. Project Folders Structure . . . . .	69
4.2. Tests and Results . . . . .	70
4.2.1. <i>ARX</i> vs. <i>Web Anonymizer</i> Performance . . . . .	72

<b>Risk Analysis as a Service</b>	<b>74</b>
5.1. Purposes . . . . .	75
5.2. <i>Re-identification</i> Test Procedure . . . . .	77
5.3. Results . . . . .	80
<b>Conclusion and Future Work</b>	<b>84</b>
Conclusion . . . . .	85
Future Work . . . . .	87
<b>References</b>	<b>90</b>
<b>Appendix</b>	<b>103</b>
A API Documentation . . . . .	103

# List of Figures

Figure 1: <i>ARX</i> Configuration Interface . . . . .	28
Figure 2: <i>ARX</i> Risk Analysis Interface . . . . .	29
Figure 3: CAT Interface . . . . .	30
Figure 4: <i>Web Anonymizer</i> Architecture . . . . .	41
Figure 5: <i>Web Anonymizer</i> Conceptual Model . . . . .	44
Figure 6: <i>Web Anonymizer</i> Mockup . . . . .	45
Figure 7: <i>Web Anonymizer</i> Interface . . . . .	46
Figure 8: Upload AJAX Code . . . . .	49
Figure 9: <i>Anonymization</i> Process . . . . .	55
Figure 10: "ARXConfiguration" Code Sample . . . . .	58
Figure 11: "ARXMainAnonymizer" Code Sample . . . . .	60
Figure 12: "mainController" Code Sample . . . . .	62
Figure 13: Home Page Interface . . . . .	64
Figure 14: Uploaded <i>Dataset</i> Interface . . . . .	65
Figure 16: Hierarchies Configuration View . . . . .	66
Figure 17: Results View . . . . .	67
Figure 18: Risk Analysis Menu . . . . .	68
Figure 19: Project Folder Structure . . . . .	69
Figure 20: <i>Ngrok</i> Tunnelling to <i>Tomcat</i> Server . . . . .	71
Figure 21: <i>ARX/Anonymization</i> Platform Performance Comparison . . . . .	73
Figure 22: <i>Narayanan et al. Re-identification</i> Conclusions . . . . .	80
Figure 23: <i>Re-identification</i> Probability Results . . . . .	81

# List of Tables

Table 1: <i>k-anonymity</i> Performances . . . . .	35
Table 2: <i>l-diversity</i> Performances . . . . .	36
Table 3: <i>t-closeness</i> Performances . . . . .	37



# Acronyms

**AJAX** - Asynchronous JavaScript and XML

**API** - Application Programming Interface

**CAT** - Cornell Anonymization Toolkit

**CNPD** - Comissão Nacional de Protecção de Dados (National Commission for Data Protection)

**CSS** - Cascading Style Sheets

**CSV** - Comma-Separated Values

**DPO** - Data Protection Officer

**EU** - European Union

**FTC** - Federal Trade Commission

**GUI** - Graphical User Interface

**HIPAA** - Health Insurance Portability and Accountability Act

**HTML** - HyperText Markup Language

**HTTP** - Hypertext Transfer Protocol

**IoT** - Internet of Things

**JDK** - Java Development Kit

**JSON** - JavaScript Object Notation

**PHI** - Protected Health Information

**PII** - Personally Identifiable Information

**QID** - Quasi-Identifier

**RDF** - Resource Description Framework

**REST** - Representational State Transfer

**SA** - Sensitive Attribute

**SDC** - Statical Disclosure Control

**SQL** - Structured Query Language

**URL** - Uniform Resource Locator

**US** - United States

**XML** - Extensible Markup Language



# Introduction

## 1.1. Context

In a digital world composed of personalized data services, there is a huge demand for data, especially for personal data. This may take the form of financial data, health data, Internet transactions, or even data based on GPS location. Large volumes of data can now be employed quite efficiently to gain new insights on a certain phenomenon, and data mining algorithms can give a lot of information about persons, events or entities. The purposes for which one requires meaningful personal data are very diversified, for example, research or public health policy purposes, to develop new services and products, to enhance the efficiency and effectiveness of new drugs, or even to simply condition people's behavior when they make a purchase on an online store. Access to data also promotes transparency and contributes to public security, and can also provide the means to ensure accountability in government and public agencies.

In recent years, escalation of technology led to an increase in the capability to record and store personal data about consumers and individuals. With this information almost anyone can track or know more about a person's life. This raised concerns on personal data misuses in many different ways. To mitigate these issues, some de-identification methodologies have recently been proposed that, in some well controlled circumstances, allow for the re-use of personal data in privacy-preserving ways. Arguably, personal data de-identification techniques, when associated with appropriate risk analysis and a comprehensive Privacy Impact Assessment, can securely unlock, in a privacy aware way, the potential of personal data, not only for research but also for commercial purposes. However, there are laws and civil rights that specifically ensure the right to personal privacy that need to be addressed, when personal data is thus de-identified and used for secondary uses that are different from the original use. However, the potential for research and commercial application of these de-identified datasets is huge, more so for example on domains like Healthcare, where the speed of scientific progress that can save lives, is often hindered by

the difficulties the research community has on collecting meaningful de-identified datasets on new drug treatment outcomes [1].

The safeguard of all personal data is highly important, however Protected Health Information (PHI) is particularly sensitive. It comprises the most sensitive and intimate details of someone's life, such as, those relating to physical or mental health, and individual's family health history. Obviously, is important to understand that protecting health information is crucial, guaranteeing the confidentiality of personal data and the privacy of the individual to whom is related. However, this information must be precise, complete, and available to all the healthcare professionals in order to provide medical care to those in needs, furthermore, all the health-related data is essential for a better healthcare system. There are other risk activities considered secondary, that also handles private data, such as, health research or the management of publicly funded healthcare systems [2]. It is unquestionable that providing greater access to data will bring many benefits to society, therefore, the question that needs to be addressed is how to make health data more accessible in a responsible way, protecting the privacy of patients, remaining compliant with current legislation and regulations, all that ideally without losing precious information about the patients.

Lately, a growing number of misuse cases with personal data resulted in a reviewing of data privacy protection regulations by many governments across the globe. It exists a European regulation, the European Data Protection Directive. In the U.S., the data privacy landscape is more chaotic and business oriented. These regulations forces the protection of critical data involving Personally Identifiable Information (PII), and PHI from unauthorized personnel. This includes application developers, testers, and any other unauthorized users by employment that have access to sensitive data. The need to comply with these regulations along with the risk of huge fines, in the case of mistreatment of customers, partners, and employees personal data by insiders, have led companies to rethink on their data privacy protection policies, and starting implementing solutions such as *de-identification* and *anonymization*.

Data *anonymization* ensures that even if *de-identified* data is stolen, it is very hard to *re-identify* it. A lot of PHI is collected, generated, stored, or transmitted by various healthcare institutions and professionals. That includes past, present or future health information of an individual

that may point to physical or mental health problems, also, this information can directly or indirectly identify a person [3]. *De-identification* of personal data is an efficient way to protect the privacy of patients when their data are used or disclosed. There are many ways to share health information without exposing the patient privacy, however, many times *de-identification* is not considered the main approach to share data in a secure way due to a lack of legal and practical reasons. Another big concern is the technology for big data privacy, a debatable question is if that technology is enough. The legal obligations imposed by data protection laws in the European Union (EU) need to be fully implemented, only this way the privacy protection will be assured. The EU committee created a research program for big data, this project includes the integration of privacy enhancing technologies as a main objective, another concern in this program, linked to the previous one, is the unstoppable constant growing of big data [4].

In this dissertation, we will present the *Web Anonymizer*, a *web* platform that provides an easy way to *de-identify* a *dataset* over the Internet as a service. The *web* application helps the user finding the better trade-off that satisfies the end-objective in terms of utility and security, by mimicking as much as possible the *ARX* desktop application on which it is based. We also conducted a well-known *re-identification* attack on a *de-identified dataset*, to prove the tool effectiveness. Through this document, the development process, technologies used and the rationale behind the choices made will be explained, hoping that in the end, the reader understands what was done here to improve this process.

## 1.2. Motivation

In data storage, between the millions and millions of records saved, there is always some information that could arm an individual in a direct or indirect way. The demand for tools that efficiently protects the personal data is enormous, mainly on real-time application, a subject still in an earlier development. Most of the mechanisms to achieve *anonymization* are private software, that uses complex algorithms usually not made publicly available [5]. The solutions that are free suffers mostly of poor graphic interface and very complex configurations processes.

Daily, the data flow coming from all the health institutions creates a massive data production that needs (or should) to be *anonymized*, of course that represents a huge investment in terms of complexity and economics [6], but entities have to realize that they are dealing with sensitive data, so actions need to be taken.

### 1.3. Objectives

The purpose of this work is to create a *web* platform that allows the user to upload a *dataset*, set up a *de-identification* configuration that suits his needs, after that, the original *dataset* and a *de-identified* version will be presented, also, at the end of the process, a *re-identification* risk analysis is displayed to the user so he can iterate again all over the de-identification process, should the results obtained are not fully satisfactory.

An important task in this project was finding the appropriate Application Programming Interface (API) to serve as a base for the *web* application. Tests were performed to search for the most suitable solutions to perform *de-identification* on *datasets*. Nonetheless, the main objective here was to secure PHI, using security procedures and protocols, avoiding major transformations on data. A frequent difficult decision that needs to be taken in account when handling PHI, is how much *de-identification* to apply (too much can turn the data useless, not enough turns it easy to *re-identify*), finding the right trade-off can guarantee security and data quality. Besides finding that trade-off, creating a friendly user platform was essential, this is important because the client will go through several configurations that could be confusing, however, these are needed in order to perform the *de-identification* process.

Knowing how to manage risks on *datasets* is crucial, a well done analysis and a good *de-identification* protects private data, such as, patient or client information. BDA offers great opportunities, e.g., reducing expenses, saving time, increasing patient health and modernizing precision medicine. Surprisingly, BDA is very recent and devalued, this approach could be very attractive in economic terms, but also, in data mining opportunities. Another concern was finding a suitable framework to create a *web* application, it had to offer features that not only facilitates the imple-

mentation, but also supports the technologies commonly used in this type of application. It is an important decision because all the project relies on it, that is why we choose the *Spring Boot*. So that framework had to be stable, easy to configure and should include all the *web* development mechanisms, such as, an embedded server or database connection.

Finally, a series of test were performed to evaluate the vulnerability of *anonymized dataset* produced by the platform, also, a successfully proven *re-identification* technique was replicated, offering a term of comparison with the software chosen to serve as core of the application.



# Background

## 2.1. *De-identification*

The terms *de-identification*, *anonymization* and *pseudonymization* are techniques used to reduce the probability of identifying persons in an unsecured *dataset*. The most common application of these techniques is on the healthcare environment, they are used to protect patient information, but they can also be applied to protect healthcare professionals, devices or institutions [7]. *Anonymization* and *pseudonymization* are two types of *de-identification* [8], they are often misused because the definition of these terms is sometimes misunderstood due to the lack of clear definitions.

The *anonymization* process has many steps, the first one is finding a trade-off between privacy and data utility, i.e., transforming data to be disclosed without being concerned about exhibiting personal information (data protection), at the same time, preserving as much information as possible (data utility) [9]. This risk mitigation is essential, so, is crucial knowing very precisely the desired end result in order to balance the two characteristics. There is a person, Data Protection Officer (DPO), in charge of evaluating the risk of data disclosure, but he also needs to take in account utility. For example, a *dataset* satisfies a given privacy specification where the utility cannot be lowered (*minimal anonymous*), in contrast to, a *dataset* fulfilling privacy requirements and containing the highest quantity of information (*optimal anonymous*) [10].

Another early step in the *de-identification* procedure is categorizing the information, there are two types of PII data, *direct identifiers* and *Quasi-Identifier* (QIDs). The *direct identifiers*, also known as *identifiers*, in a *dataset* are those fields that can be directly used to uniquely identify individuals or their families, e.g., name, telephone number, social security number, healthcare number or email addresses. The QIDs (also known as *indirect identifiers*) are fields in the *dataset* that can be used to identify individuals but not in a direct way, e.g., dates, ZIP codes, city, state or facility names [2] [11]. Generally, is applied the distinction between these two types of identifiers

because they need to be treated different manner [12], there are also two other types of attributes, sensitive attribute (SA) and (*insensitive*), but some publications don't consider them as a type of data. The last step is the *de-identifying* itself, there are several techniques to do this, in the *Privacy Models* section those methods will be presented and explained in more detail.

### 2.1.1. *Anonymization*

Anonymity, means simply that a person is not identifiable. The *anonymization* process is intended to irreversibly remove the association between an individual and some information that can identify this person. If the process is meant to be reversible and a certain identifier replaces the person's real attribute, then this procedure is called *pseudonymization* [7].

Researchers have developed a scoring system with various *anonymization* techniques, they vary in cost, complexity and robustness. The most common techniques are: *generalization*, *suppression*, *micro-aggregation* and *subsampling* [14] [96], will talk more about them in the *Privacy Models* section. The opposite of *anonymization* is *re-identification*, or could be found in some literature as *de-anonymization*.

As seen before, personal data is *anonymized* to protect the privacy of subjects when storing or disclosing data. Knowing how the data will be disclosed is important to the person who makes the *anonymization* process, generally, data is disclosed in three ways [15, 16]. First, releasing data to third parties (e.g., data analyst sharing information with another analyst). Second, administrators occasionally disclose *anonymized* data to the public [17]. Third, administrators release *anonymized* data to others within their company, that happen particularly in large organizations [18]. One thing is for sure, every (or most of) professionals agree that *anonymization*, in contexts such as e-commerce, data mining, national security or academic researches, is essential to protect human private lives. Furthermore, professional statisticians are duty-bound to *anonymize* data as a matter of professional ethics [19].

Many defend the privacy-protecting power of *anonymization* and hold it out as a best practice, but many times they don't apply it due to costs and time. Many legal scholars and analysts share this faith in *anonymization*, likewise, *Google* following United States (US) and EU recom-

mendations, was the first search engine to announce that they will *anonymize* their search logs and do not keep *personal identifiers* in their logs for more than six months [20].

### 2.1.2. Pseudonymization

The term *pseudonymization* or *pseudo-anonymization* refers to a particular type of *de-identification* that removes the association between data and a person, introducing a new identifier that establishes a bidirectional-mapping between the individual and his identifier. Pseudonymous data is still considered personal information under the European Data Protection Directive 95/46/EC [21], and for that, should not be considered as anonymous. Until now, all known *re-identification* attacks were performed on pseudonymous data [22, 23], that happens due the weak *de-identification* achieved by this technique in comparison to *anonymization*. *Pseudonymization* is a peculiar type of *de-identification*, it removes the association with a PII, adds a connection between one or more pseudonyms and a certain set of characteristics related to the personal data. This method is divided in two different processes, irreversible and reversible *pseudonymization*. In the first one, the *pseudonymized* data don't keep information that allows re-establishing the association between *pseudonymized* data and the person's data. It adds anonymity, but the pseudonym continuity is protected on the produced *dataset*. In the reversible one, *pseudonymized* data can be related to the personal data by applying measures restricted to only authorized users.

A *pseudonymized* database must contain at least two tables, one will store all the personal data, the other saves pseudonyms and *pseudonymized* data. *Pseudonymity* is an approach that provides a form of traceable anonymity and requires legal, organizational or technical procedures, consequently the association can only be accomplished under specified and controlled circumstances [24, 25].

## 2.2. Re-identification

*Re-identification* is the reverse process of *de-identification*. The number of *re-identification* attacks on private *datasets* has grown a lot in the last few years. Experienced professionals

realized that the removal of direct identifiers could not insure a properly *de-identified dataset* [26], so they created an ability to measure the probability of *re-identification*. Various metrics have been established to classify this probability [2], these parameters can be applied on *datasets* or simply on personal information. The probability of *re-identification* will depend on two elements: which QID are included in the *dataset* and what was the degree of disturbance the data has suffered. In general, the more QID are in the released data, the easier will be to *re-identify* the information [23].

Another issue to have in account is the disclosure of information, for non-public data releases is fundamental to not forget the possibility of a person attempt to *re-identify* an individual in the *dataset*. Considering attacks on PHI, some guidance and standards [21] [26–28] were recently released, also, a scheme based on subjective probability has been used to classify the chances of *re-identifying* personal data, this program returns a probabilistic value based on the *re-identification* risk [2]. This framework performs some checks on the data recipient, the security in place, the motivation (technical and financial) and the ability to *re-identify* the *dataset* [12].

Furthermore, *re-identification* risk is defined by the prospect of positive matches, the number of verification attempts made by the attacker will measure the risk, this menace can be controlled at all levels if the attacker's motivation is known [29].

## 2.3. Data Protection Officer

Recognizing the critical influence of the DPO role is mandatory for data privacy corporate accountability, recently, many companies have invested in hiring or training a DPO. Few countries currently force the appointment of a DPO, yet there has been a growth in the number of DPOs appointed. The EU Regulation [28] orders the appointment of a DPO in various situations, and establish the particular functions and responsibilities of this professional. The exact criteria for assignment will depend on the current version of the regulation, EU commission recommends appointments based on the number of employees a data controller has, or on where its personal data processing activities are located. Also, the EU parliament proposes assigning a DPO related to the number of affected data subjects or core activities a company has [30]. The EU Regu-

lation defines the tasks and responsibilities of a DPO [28], that includes: providing information and awareness; advisory function; overseeing and monitoring data protection; handling queries and complaints; maintaining documentation; consulting and cooperating with regulators; dealing with data subjects directly and organizational functions. Specific tasks include conducting audits, monitoring data protection impact assessments, managing the implementation of data protection by design and default, developing staff training and ensuring data security [31, 32].

When the EU Regulation is infringed, a company/person risks heavy fines from the regulatory authorities. This fines could reach a €1 million or 2% of the annual worldwide turnover of a business per violation. Besides, personal information privacy is vital to the trust that customers have on a company, when a breach becomes public it reflects on serious reputation damage to the business. A DPO can really help a company, he will reveal and manage risks in an early stage, avoiding catastrophic disasters in both financial and reputation terms [33].

In the particular case of a *de-identification* framework, the DPO will tell what is the degree of risk according to a series of parameters. Based on his assessment, the data will be *de-identified*, then the DPO will test the *anonymized dataset*, with the help of existing programs and frameworks, trying to *re-identify* the information. If tests results were not satisfying, other *anonymization* methods will be applied in order to *de-identify* the data properly, otherwise, he should warn the entity to not disclose that information.

## 2.4. Legislation

Many countries around the world treats privacy in their own way, some countries apply more restrict laws, others not so much, and there are those that don't have a legislation at all. In an important matter such as privacy, we though that will be interesting to see how some countries act in relation to personal information, we will also present a comparison between EU and USA, the two regions that takes this subject more seriously, and also they currently hold the oldest and thorough legislations in the world.

### 2.4.1. Worldwide Data Privacy Laws

- **People's Republic of China**

China has the biggest number of Internet users, approximately 500 millions. Privacy is a recent right in the Chinese legislation, and some of their concepts about privacy are not recognized by the occidental society. Before announcing "Law of Torts" [34, 35], in 2009, the Chinese legislation (Constitution and General Principles of Civil Law) didn't recognize privacy as an individual right. However, other protection laws exist to defend the personal privacy, such as, the Articles 38, 39 and 40 of the Chinese constitution [36], granting dignity, residency and confidentiality. There is also the Article 253 [37], that protects the citizens of undesired disclosure of private personal information by governmental employees and certain companies operating in financial, health, telecommunications, education or transportation domains.

In May 2012, the National People's Congress has created a law proposition aiming cyber security, where different types of personal information are supervised by particular laws and regulations [38]. The Measures for Punishment of Infringements on Consumer Rights and Interests announced by the Congress, defines consumer's personal information as "the information collected by business operators during the provision of goods or services that may be used for identifying a consumer either independently or in combination with other information, and shall include name, gender, occupation, date of birth, identity document number, residential address, contact details, income and asset conditions, health conditions and consumption habits of a consumer" [39]. Moreover, the National People's Congress defines PHI as all information containing demographic data, medical and health-care services information and other population health data.

The PHI is created by all types of medical healthcare, family planning services and during the process of providing information to laws and regulations. Furthermore, the national law requires that all the PII related to Chinese citizens have to be stored on servers located in China, and those need to be verified periodically by national entities.

- **Russia**

Around 2006, Russia has redacted for the first time a law to protect personal data, but this document didn't specify the accountable agency for the supervision of this law, so, many entities used their power in self benefit, making harder to understand which agency had the main role. Soon after, the Federal Service for Supervision of Communications, Information Technology and Mass Media had been nominated as the responsible agency to ensure the compliance of data protection law, overwriting in power all the other entities [40].

The Russian law, similar to the EU, imposes a series of restriction on using, storing and processing personal data. The law defines "personal data" as all the information related to a person that can be identified through that same data, this information may be a name, address, financial and social data, education, occupation or even salary [41]. In Russia, the companies are allowed to collect, use, save, process and share PII only if the purpose is in accordance with established law or with a written consent by the data owner. This policy will prevent undesired disclosures of personal data, but there are others measures recommended, such as, defining personal data categories, asking for consent, appointing a DPO, adopting data protection policies and locating data servers in Russian territory [42,43].

The law assures that patients have freedom of choice in terms of medics, healthcare institutions and insurance policies (Article 6) [44]. Moreover, "insurance companies are not part of the healthcare system" [45] and "government bodies supervising the healthcare system and medical entities can't set up health insurance companies" (Article 14) [44].

- **India**

India is in this analysis group for two reasons, first, cause India is the second country with more people connected to the Internet (behind China) [46], and because the level of regulation and data privacy enforcement is considered low, instead of the EU (high) or China (moderate) regulations [47].

In 2000, the Indian government redacted an Information Technology Act, called "IT Act", containing recommendations intended to protect electronic data (including all non-

electronic records). Later, in 2011, the government has published the Information Technology Rules (IT Rules), addressing security procedures and specifies some basic rules on personal privacy. The Section 43A of IT Act, describes what is sensitive personal data, e.g., passwords, sexual orientation, medical records, biometric and financial information.

In the IT Rules act, is explained what kind of actions related to sensitive personal data are regulated by this law, e.g., collection; receipt; possession; use; storage; dealing or handling; transfer or disclosure; security procedures; review and correction; transferring outside India and erasing without consent. When data is collected, the entity responsible is obliged by law to inform the person in question what is the purpose of that collection, who will gather, store and analyse the data and when will be collected. However, notification or registration is not required before processing data, i.e., the PII can be reused as much as they want in the future without any consent [48]. In this act, nothing is specified in relation to PHI, all sensitive data is handled in the same way.

#### **2.4.2. EU vs. USA**

Under Article 8 of the European Convention on Human Rights, the right to protection against the compilation and use of personal data forms part of the right to respect for private family life, home and correspondence. A person has the right to protection against intrusion from others, especially from the state. Used for the first time as a legal instrument in Article 12 of the United Nations Universal Declaration of Human Rights of 1948, on respect for private and family life [49]. All the European state member had now introduced or make an effort to the implementation of Human Rights in their national law, which requires them to act in accordance with provisions of the Convention. In 1981, a conference for the protection of individuals regarding the automatic processing of personal data as occurred, there the Convention 108 [50] was redacted, and still remains the only legally unifying international instrument in the data protection field.

The main tool on data protection is the Directive 95/46/EC of the European Parliament and the Council, redacted on 1995, protecting the processing of personal data and the free circulation of such data [51]. This directive had the goal to ensure the same level of protection rights and



freedoms of individuals with concern to the management of personal data in all the countries. Later, a law regarding the processing of personal data by institutions in the EU and the free movement of such data was created (Regulation (EC) No. 45/2001) [28]. The personal data definition is very similar to the ones explained previously in the others legislations, but to the EU interpretation is added that: *anonymized* data don't present any more PII, *pseudonymized* data has all the PII encrypted, and *pseudonymized* data is still personal data [52]. The EU law defines a special category of personal data, due to its nature (racial or ethnic origin, political, religious, health or sexual life) can arise risks to an individual. This sensitive data must be handled with care, requiring active measures by controllers to encourage data protection in their processing actions, and be accountable in complying the data protection law in their processing activities.

Electronic health file systems are implanted in most of EU countries, but using this technology without complying with the rules for processing the data should not possible, based on Article 8 of the Data Protection Directive. PHIs are qualified as sensitive data under this Article and by the Article 6 of Convention 108. The processing of health data is allowed for preventative medical purposes, such as, diagnosis, provision of care, treatment or management of health-care services. Processing is allowed, however, must be done by a healthcare specialist under professional secrecy, or by another competent person with equivalent obligations.

The *pseudonymization* could be the solution needed to satisfy scientific demand and at the same time protect PHI, of course, there is some doubt about this method, opening intensive discussion worldwide about how to store electronic health files [53]. In the EU, these digital systems are being studied in order to make them available across borders [54], but many other initiatives are being object of research, mainly regarding personal electronic information in the health sector [55].

In the US, there are no major federal law regulating the collection and use of personal data. Instead, there is a series of federal and state laws, guidelines and regulations, that frequently overlaps or contradicts on each other [56]. Furthermore, governmental agencies and industrial corporations disclose some regulatory guidelines and frameworks that are considered "best practices", but they don't have the power of a national regulator. There is a wide range of federal

privacy laws that regulate the collection and use of sensitive data, two of them are very popular, the Federal Trade Commission (FTC) act and Health Insurance Portability and Accountability Act (HIPAA). Also, the National Institute of Standards and Technology (*NIST*) had developed some interesting studies on *de-identification* and *re-identification* techniques, these focus mainly on protecting the person's privacy, so, these mechanisms could be important to US government agencies [57,58].

The FTC Act is a more general law, applied to most institutions and individuals doing business in the US, other than certain transportation, telecommunications and financial companies (these industries are supervised by other national regulators). The FTC Act does not regulate a particular kind of data, rather, prohibits deceptive acts or activities involving the failure to protect personal information [59]. The HIPAA, released in 1996, regulate covered entities and business associates, this includes health plans, medical institutions and healthcare providers who makes business over electronic transactions [60]. A covered associate is a person or entity that executes certain activities involving the process or disclosure of PHI on their behalf or to another covered entity [61]. More precisely, the HIPAA regulates PHI, which corresponds to supervision of collecting, storing and transmitting electronic personal identifiable health information between covered institutions. So, they added a privacy rule regarding methods for *de-identification* of PHI, these techniques presented were called "Safe Harbor" and "Expert Determination". "Safe Harbor" focus on eighteen different types of information, which can help in *re-identifying* personal data [8]. Sixteen of them are considered as identifiers and include attributes such as name, telephone number and social security number. The other two are known as QID, including date and location. The hope is that by changing or eliminating PHI, the patient's identity cannot be traced back to an original *dataset* [62]. The Expert Determination method handles both direct and indirect identifiers, and the main objective is calculating the risk and managing it. This technique guarantees both the need to protect the person's identity, while allowing corporations deep analysis on data. The process requires a person with knowledge on statistical, scientific principles and techniques for rendering information. The applications of such standards and methods, determines if the *re-identification* risk is very small and the information is still useful [62].

The HIPAA requires entities to obtain written consent from a data subject before processing, these consents must contain the signature of the individual and the date. Currently, a person cannot request the deletion of their information under applicable federal laws, but through HIPAA, he can demand that incorrect or incomplete information is rectified, however, the controllers are not obliged by law to do such thing. There is also a guide for the remote use and access of PHI that specifically discuss the risks related to storing, accessing and transferring medical data on computers, wireless devices, flash drives and email [63]. The HIPAA requires entities to notify individuals when their PHI has been breached, also they are obliged to alert the Secretary of the US Department of Health and Human Services in less than sixty days, and finally, they are duty-bound to notify the media if a breach occurred involving more than five hundred individual [64].

In 2013, HIPAA suffered a reform, the "Omnibus Rule", a movement to strengthen personal privacy while continuing to regulate others interests, namely public healthcare, in greater access to health information [65]. This rule expands the meaning of a "covered business" to include all companies that create, collect, store, or transmit PHI on behalf of a covered entity, making clear that controllers storing PHI on behalf of healthcare providers are business associates. The HIPAA Privacy Rule generally forbids the process or disclosure of PHI for marketing purposes without personal consent, but there was certain exceptions. The "Omnibus Rule" came to reverse this situation, PHI is no longer available to be used in marketing action without the subject's authorization. Another situation rectified, is the case of PHI sale without consent. The "Omnibus Rule" specifies in the case of violations, penalties that could reach \$1.5 million per violation. To apply these fines is accountable the number of persons affected by the data breach, the weight of non-compliance and the severity of negligence by the company [65].

As technology quickly advances, PHI is collected and saved in many environments, and HIPAA's goal in protecting health information is harder to achieve, that happens because data is protected mostly on when and by whom was collected [66].

The major differences between the two legislations are, the excessive number of laws and the lack of a regulatory entity in the US. This evident panoply of laws as a negative effect on data protection, there are too many laws, regulation and guidelines controlled by too many agencies.

In the healthcare sector, this chaos is not that perceptible, because of HIPAA, a detailed and well structured act in comparison to other activities laws. The US legislation has laws to protect personal information, but the privacy stays at the companies charge, and most of the time it doesn't matter how is achieved. It is notable that the companies try to defend themselves instead of being concerned with individual's privacy, they are more worried about sanctions, and in case of disclosure, a trial that could be really hard financially and bad for the company's reputation. In 2016, the US Congress enacted the Judicial Redress Act, allowing citizens from allied nations the right to pursuit atonement in US courts for personal data violation by law enforcement agencies [67], this made the US legislation came a little bit closer to the EU regulations.

#### **2.4.2.1. European Data Protection Regulation**

In May 2016, the European Parliament approved a new Data Protection Regulation, published in the Official Journal of EU, under the name of Regulation 2016/679 [21]. This law is taking a two year transitional period, until May 2018, giving the entities time to adapt the new legislation. Being a European regulation, overrides any national laws, guaranteeing homogeneity between all the EU members. This new guideline includes several upgrades to the previous Data Protection law (Directive 95/46/CE), particularly forcing corporations to comply with privacy protection, otherwise, high penalties will be applied, such as, 20.000.000€ or up to 4% of the company's annual profit. In terms of structural organization, this new regulation introduces accountability duties, imposes regular privacy impact assessments, notifying the responsible authorities in case of data breaches, assigning a DPO and reinforcing data security.

This document, also defines the concept of personal data, providing new rights to data owners, e.g., right to be forgotten, right to data portability and the right to object profiling. Furthermore, obtaining the owner's consent is absolutely necessary and a far more strict process. Another focus is the data treatment, but new principles and approaches are presented, focusing mainly on privacy by design and by default, also, the old directive only was applied to controllers and not to processors. Besides, the law will manage the processing on European citizens, regardless the localization of the responsible controller/processor being inside or outside EU territory.

Finally, the transitional period will give organizations time to review their processes and policies, so, until May 2018, they need to adapt themselves in order to comply with the new legislation, avoiding this way heavy penalties and sanctions.

### **2.4.3. Portuguese Legislation**

Being Portugal a Member of the EU, the national data protection laws are very similar to the EU regulations. The main law is the act 67/98, which derives of the EU Directive 95/46/EC of the European Parliament and of the Council, on the protection of individuals with regard to the processing of personal data and on the free movement of such data [68]. There are other laws concerning the processing of personal data in the context of publicly available electronic communications networks and services, law 41/2004, which implemented the EU Directive 2002/58/EC. Another law (32/2008), implementing the EU Directive 2006/24/EC of the European Parliament and Council, on the retention of data generated or processed in connection with the provision of publicly available electronic communications services or public communications networks [69]. Finally, as a member of the *Schengen* space, there is a law (Act 2/94) that establishes the control and verification mechanisms for the information system, "Schengen Information System has the purpose of preserving public order and security, including the State security, as well as applying the Convention provisions on the circulation of persons in the Contracting Parties territories, with the support of information transmitted through this System" [70].

In the Portuguese legislation, personal data means any type of information (including sounds and images) that can identify or helps to identify a person. Moreover, sensitive data is defined by all personal data concerning philosophical or political beliefs, religion, private life, racial or ethnic origin, political party or syndicate membership and health or sex life, including genetic data [71]. The "Data Protection" law instructs the notification to the data protection authority, *Comissão Nacional de Protecção de Dados* (CNPD), the regulatory entity in Portugal, before performing processing on sensitive data, financial data, combination of various personal data or transferring data to foreign countries. In order to analyse personal data, is required to obtain prior consent with the original data subject, and that must contain the personal data to be collected, the purposes of

the collection, the data controller's identity, the right of access, modification, elimination, sharing and the countries where the data may be transferred to. There is no specific rule related to minors information, however, children under eighteen years of age have limited legal decisions and therefore is needed to evaluate if the consent is within the scope of their admissible capacity, otherwise, a child's legal representative must provide the consent.

The treatment of data related to health, including genetic data, is authorized when serving for preventative medicine, medical diagnosis, provision of care or treatment and management of healthcare services. In a situation that a health professional needs PHI, under the national law, the CNPD must be notified and all the measures to protect that information must be taken by the professionals [72]. Data subject's have a legal right to request the deletion of their data, also implied by the regulation of EU Regulation (EC) 45/2001 (Article 17) [28]. If such a request is made, the data owner must comply promptly and without any costs or imposition. The CNPD can impose fines if the data holder negligently fails to the obligation of notifying this entity about processing personal data or provides false information. The fines for misusing one individual personal data can go from 249,40€ up to 14.963,94€, but these values can be doubled to an approximate cost of 30.000€, if the data processed were sensitive data and under CNPD's authorization. In the communications sector, breaching certain rules is also treated as an administrative offence, resulting in fines up to 5 million euros when committed by a legal person [69].

## **2.5. Personal Information Breaches**

Here, we will present two major cases where personal information was disclosed without consent, when this occurs millions of persons could be affected, and sometimes they don't even know that company hold their personal information.

### **2.5.1. Netflix**

In 2006, *Netflix*, the "world's largest online rental service", released one hundred million records publicly, revealing that nearly a half-million of its users had rated 17.700 movies [73]. Each

record contained a client ID, the rating given and the date of that evaluation. *Netflix* anonymized the records, removing identifying information like usernames, but assigning a unique user identifier to preserve the rating continuity. The company declared that they had a good reason for revealing these records, they wanted to improve their movie recommendation system.

In order to turn it accurate, *Netflix* released the records and throw a data mining contest called the "Netflix Prize", that took three years to claim [17]. The first team that would improve significantly the recommendation algorithm would win one million dollars [74]. Thousands of researchers have competed for this prize, not only for the money, but also to create or improve statistical theories [75]. If an adversary, the term used by computer scientists, knows the precise classifications that a client made to six movies in the same category, identifying that person will be possible 84% of the times [76]. If he knows approximately when (two weeks error) a client made six ratings, whether or not they are in the same genre, is possible to identify the person 99% of the times. Later in 2009, some *Netflix* clients have initiated a lawsuit, they claimed that the company disclosure of information for their contest had violated personal data privacy. In the case, they alleged violations of several state and federal privacy laws, later on, *Netflix* declared that had settled the suit and the plan for a second contest was dropped [15].

### **2.5.2. TRICARE**

In 2011, the biggest data breach incident reported to federal regulators under the HIPAA breach notification rule was observed. An unencrypted backup computer containing electronic record tapes from the health system, including military retirees, as well as active-duty troops and their dependants records, in about 4.9 million Military Health System patients have been breached by a contractor for the *TRICARE* insurance carrier [77]. The breach was reported to *TRICARE* on Sept. 14, involving backup tapes with military hospitals, clinics and pharmacies information, from 1992 in the San Antonio area. *TRICARE* informed that the breached records may include social security numbers, addresses, phone numbers and some personal health data, such as, clinical notes, laboratory tests and prescriptions, but did not hold any financial information [78]. The San Antonio police reported that the tapes were stolen from an employee's car during

a burglary. The contractor was accused of failing to maintain standard procedures in order to protect *TRICARE* clients personal information, also, they were accused of violating state laws by failing to notify the authorities of the theft [79]. *TRICARE* defend himself saying that the risks of harm were low, despite the PHI involved. Because specific equipment was needed to read the tapes and a expert to interpret the data, meaning that the odds of accessing the data were low [80]. Moreover, *TRICARE* tried to position themselves under FTC regulation instead of HIPAA regulation. Unlike HIPAA, the FTC regulation don't demand that covered entities signs agreements with their "business associates", requiring third parties to apply the same standards when handling sensitive data [81].

"In general, is important for HIPAA covered entities to ensure that backup tapes are included in their analysis and management risk plan. If encryption is not feasible, covered entities should focus on strong administrative and physical safeguards, such as clear procedures that ensure backup tapes are locked up at all times" said Adam Greene, a former official at the Department of Health and Human Services Office for Civil Rights [82]. Later, thirty three complainants initiated eight class action suits against that company, but only two were accepted, the ones that accused the company of personal data breach [83].



# State of the Art

In this chapter we will present the technologies available to perform *de-identification* on *datasets*, but firstly, some concepts will be explained in order to understand how these solutions work. Firstly, we will talk about the most popular privacy models used nowadays, after, an explanation about hierarchies and why they are so important, finally, the *de-identification* solutions available on the market today will be presented.

## 3.1. Privacy Models

When healthcare entities disclose *datasets* containing PHI, there is always the risk of *re-identification*, some *anonymization* algorithms are based on sanitization methods, such as, *generalization* and *suppression* of QID attributes. These *anonymization* methods normally use syntactic sanitization, e.g., *k-anonymity* needs that each QID tuple occurs at least in  $k-1$  records, *l-diversity* demands that the SAs dispersion for a QID must have high entropy and *t-closeness* requires that the SAs dispersion in any equivalence class is similar to the SAs dispersion in the overall *dataset*. There is more *anonymization* algorithms, such as *m-invariance* or *delta-presence*, but we will focus on those three because, actually, they are the most popular and effective ones.

- ***k-anonymity*** - This algorithm requires that each QID tuple appear in at least  $k-1$  records, this will ensure at minimal, that released data processed with *k-anonymity* will be difficult to re-identify [95] [96]. The QIDs contains information that is more likely to find over the *dataset*, so this type of attribute is more vulnerable to *re-identification*. *k-anonymity* uses *generalization* and *suppression* methods. *Generalization* implicates replacing a value with a redundant but semantically similar value. *Suppression* involves not publishing a value at all. There are several different methodologies available, combining these two can be very powerful [97], but sometimes, *anonymization* is not enough. Imagining that a *dataset* protected with *k-anonymity*, this means that if an attacker has any identifying information

about an individual, is certain that at least  $k$  records in the *dataset* can be linked to that individual (background knowledge attack). Moreover, assuming that those records also include sensitive information, e.g., a person is diabetic. If this sensitive information is equal for all  $k$  individuals, then *k-anonymity* cannot protect that fact to be disclosed (homogeneity attack) [98].

- ***l-diversity*** - This algorithm requires a high entropy on the distribution of SAs for each QID. Overall, *l-diversity* is effective, intuitive and solve most of the *k-anonymity* failures. Besides, a trusted privacy against attacks is used, even when collectors don't have any kind of information about the attacker's level of knowledge.

The main idea behind *l-diversity*, is the well balancing dispersion of SAs between all the groups included on the *datasets* [99]. However, *l-diversity* also has vulnerabilities, such as, similarity attacks, that happens because he considers the diversity of SAs in the group, but he is not concerned with the semantic proximity of the values. Skewness attacks demonstrated another vulnerability, they happen due to the equality of positive and negative values in the second equivalence class, giving to every record in the class a 50% chance of matching, much higher percentage when compared with the real distribution [100]. The third equivalence class is the most fragile, the algorithm assumes that the attacker has zero knowledge on the global distribution of SAs, however, the attackers can learn the sensitive dispersion by just looking at the *dataset*. The general issue with *l-diversity* is the limitation of his assumptions related to the attacker's intelligence [101].

- ***t-closeness*** - Due to the fact that previous privacy models had some vulnerabilities, a new one emerged, the *t-closeness* algorithm. He requires that the distribution of a SA in any equivalence class must be similar to the attributes distribution in the overall *dataset*, this way, the chances of learning individual's information are lower. In order to introduce and manage gaps between values of SAs, *t-closeness* uses the *Earth Mover Distance* metric [102], receiving a precise distance between the two distributions. *Earth Mover Distance* measure on *t-closeness* has the advantage to take in consideration the semantic closeness

of attribute values essential on the *anonymization* process robustness. This method allows the data collector to use other *anonymization* techniques besides *generalization* and *suppression* [103]. Using other techniques is recommended rather than suppressing a whole record, because a lot of information is lost, but also, eliminating a value only reduces diversity on the *dataset*. Moreover, removing an *outlier* may turn a distribution more uniform and make it similar to the general distribution. Another technique is the *generalization* of a SA value, instead of hiding it the value is randomized.

Many studies are focus now in how to merge effectively these techniques with *generalization* and *suppression* to obtain more resistant and useful *anonymized dataset* [101].

### 3.2. Hierarchies

Defining these intervals is important in the configuration process, all the solutions below use this mechanism, but sometimes is not clear what their function is, neither the right way to configure them. So, the objective here is to explain what are these hierarchies, what are they used for and how to correctly define them.

*Generalization* hierarchies are typically used in *de-identifying datasets* with PII. Using this type of configuration will reduce the precision of attribute values, i.e., instead of having simple values, the data will be presented as intervals. Imagining an attribute containing the patient's age, from 0 to 100 years old, a hierarchical option, in this case, could be using decade intervals. This will result on ten levels, and the data will be presented as: "[0, 9]", "[10,19]" and so on. Defining bigger interval (fewer levels of hierarchies) will decrease the risk of *re-identification*, however, if this attribute is relevant for an analysis, the *dataset* will lose utility [85] for that purpose.

Hierarchies are normally used for categorical attributes, but they can also be used for continuous data, but that will demand a technique called "categorization". To increase the utility of *anonymized datasets*, "categorization" is often combined with tuple *suppression*, i.e., data records inconsistent with privacy criteria (*outliers*) are automatically removed from the *dataset*. While the number of suppressed records is kept below a given threshold, less *generalization* is needed to

guarantee that the remaining records fulfil the privacy criteria [84]. Data and *generalization* hierarchies can be imported from many different types, providing compatibility with a wide range of data processing tools. *ARX* allows importation, but also offers a system that authorizes the user to define hierarchies that suits him better, likewise, a semi-automatic creation is available for beginners or inexperienced users [104].

### 3.3. *De-identification* Solutions

Nowadays, there are some tools available in the market that transforms a *dataset* into a *de-identified* one, but mostly, these software requires installation into a machine. Some of them are open source, others are private, but none of them are *web* oriented like we intend to do. Below, a more detailed overview of these *de-identification* solution will be presented, including a little discussion about their pros and cons, also, we will see how was the user experience on maneuvering these software.

To perform this study, the released "Netflix Prize" database was used. The reason for this choice was knowing that this *dataset* was successfully *re-identified* in the past, giving us the opportunity to prove that a *anonymized dataset* produced by the *Web Anonymizer* is secure and still useful for analysis.

We will also analyse the performance of these software in terms of efficiency, execution time and user experience. This step is important for the project because the platform to be created, will have on the software with the best results its base, using their API, with the objective of building a *web* version of it.

#### 3.3.1. Open Source

- **ARX** - The *ARX* open source tool allows the user to change structured sensitive personal data, using Statical Disclosure Control (SDC), into data that can be shared securely. The objective is to reconstruct *datasets* in compliance with well-known syntactic privacy models, that will reduce the success of attacks, preventing privacy breaches [84].

*ARX* focus into PHI, being able to remove direct identifiers and introduce constraints on indirect identifiers. These QID are always assumed by the tool, as being known by the attacker, but they cannot be erased from the *datasets* because they are needed for processing. Since version 3.3, *ARX* supports semantic privacy models, providing better knowledge about the attackers experience and their intents [85]. The tool focuses on minimizing the risks of disclosure, "cleaning" the data in a way that usual vulnerabilities are reduced, guaranteeing the minor loss of information on the *dataset* processed.

*ARX* system uses utility measures only to compare the results of different data transformations to each other, both types of measures can be used together defining information loss as the opposite of utility. Furthermore, an intuitive coding model is implemented in a friendly user interface (figure 1) that gives the client an acknowledge freedom to configure the more suitable *anonymization* method, after it presents a powerful graphical visualization of the complete process, resulting *dataset* and a detailed risk analysis [85].

This software gives the user three options to upload data, *.csv*, *.xls* and Hypertext Transfer Protocol (HTTP) files, allowing the *anonymization* of almost all the data produced worldwide. There are other formats, not so popular, that this platform don't support, such as, *.tsv* (tab-separated values), *.html* and *.odb* (*OpenOffice Organizers Database*), but they can be easily normalized in a supported format with the help of a text editor. In terms of hierarchies, they are simple to build with many options on intervals, with a large number of levels, being easy to create and manage them. Furthermore, defining the attribute type only requires selecting it and choosing the type from a list, the same applies for the privacy models, the possibilities are listed and with the help of a knob the user selects a value to classify the chosen algorithm. Another measure that can be adjusted is the coding model, that could go from 100% *suppression* to absolute *generalization*, or something in between that. The *ARX* also gives the opportunity to choose one, or several, attributes and define the weight they will have, i.e., on SAs is recommended a bigger weight (more severe *anonymization*) due to the greater risk of *re-identification*.

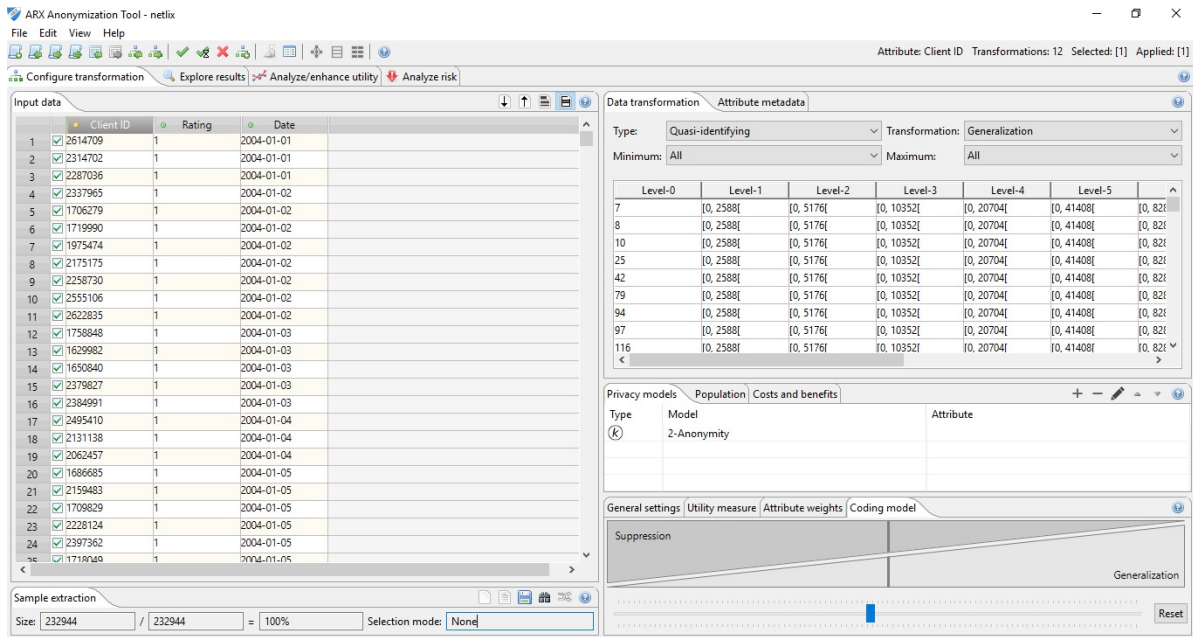


Figure 1: ARX Configuration Interface

In terms of the *anonymization* process, the user doesn't have any visual perception of the progress, however, in the worst case tested this procedure only took a few seconds, and after his completion ARX returns a thorough information of what happen. When the *anonymization* process is completed, a risk analysis view is available (figure 2), presenting a histogram and a table with the *re-identification* percentage of each attribute, or a combination of them. Another interesting tab is the "Analyse/enhance utility", that shows the original and *anonymized* data side by side, giving the user a chance to see the differences between them, but more importantly the level of anonymity achieved. The informations mentioned here are only the basics, much more risk analysis details are presented to the user, e.g., statics, distributions, contingencies and a graph that combines different solutions that can be applied to the *dataset* in order to achieve a more effective *anonymization*.

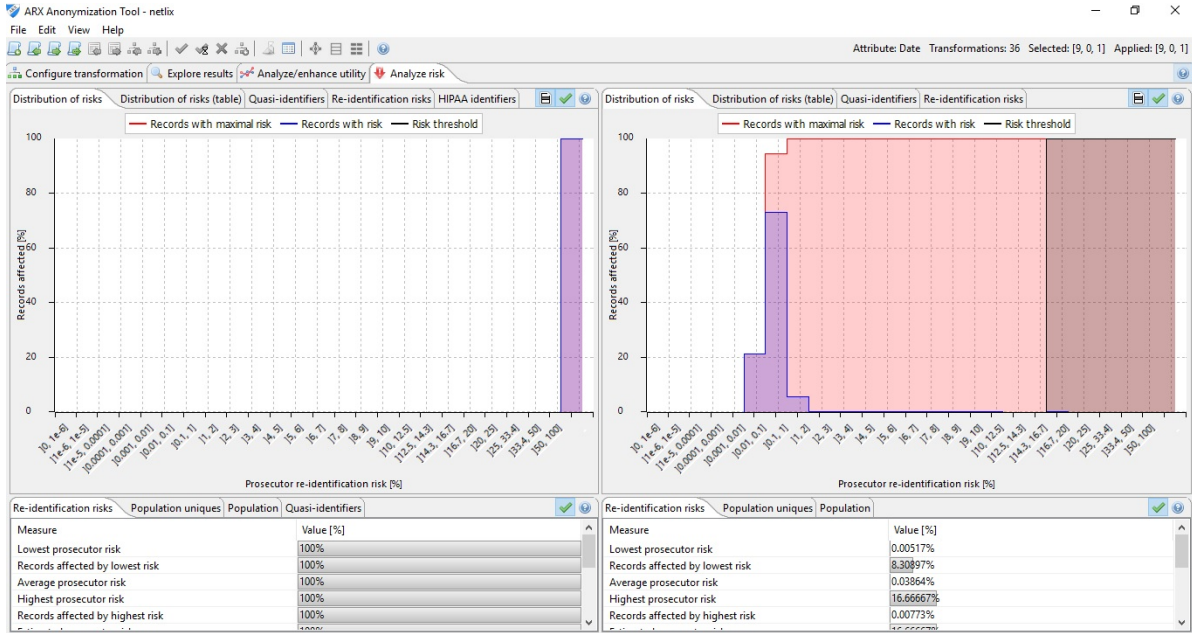


Figure 2: ARX Risk Analysis Interface

- **CAT** - This software was created with the purpose of *anonymizing datasets* in order to be published without revealing personal information, or at least reduce *re-identification* risks. Being a toolkit, it contains mechanisms that focus mostly on data generalization, risk analysis, utility assessment, SA manipulation and user interface synergy. To begin, this application requires the input to be in micro and meta data format, this can be a huge problem for enormous *datasets*, complex ones could take several days to transform in the supported format. Another difficulty found was in the hierarchies, is required to create them manually, i.e., the intervals that compose the tree need to be designed one by one, that will take another huge amount of time on the responsible analyst. The *anonymization* process, like ARX, takes only a few seconds to be completed. However, only *generalization* is allowed, and beyond that, *l-diversity* and *t-closeness* are the only models available.

Another advantage of this software is the ability to configure the *anonymization*, analyse, and reconfigure it until a satisfactory result for the user is reached. This is possible

cause the original *dataset* is kept in the main memory, and all the modifications are executed on that main-memory data, which is only outputted to disk when the users gives that order [86]. Cornell Anonymization Toolkit (CAT) presents some tools to evaluate the risk on *anonymized* data, allowing to select one or more attributes in order to check the percentage of tuples with a certain risk value. The user can define a threshold, based on the risk value percentage, and remove all the tuples outside that risk limit, this will delete the entire record of the *dataset*. The software has another three ways to compare original and *anonymized* data, a contingency table, and two graphical perspectives, joint and marginal densities (figure 3), but only allows to compare two attributes at the same time.

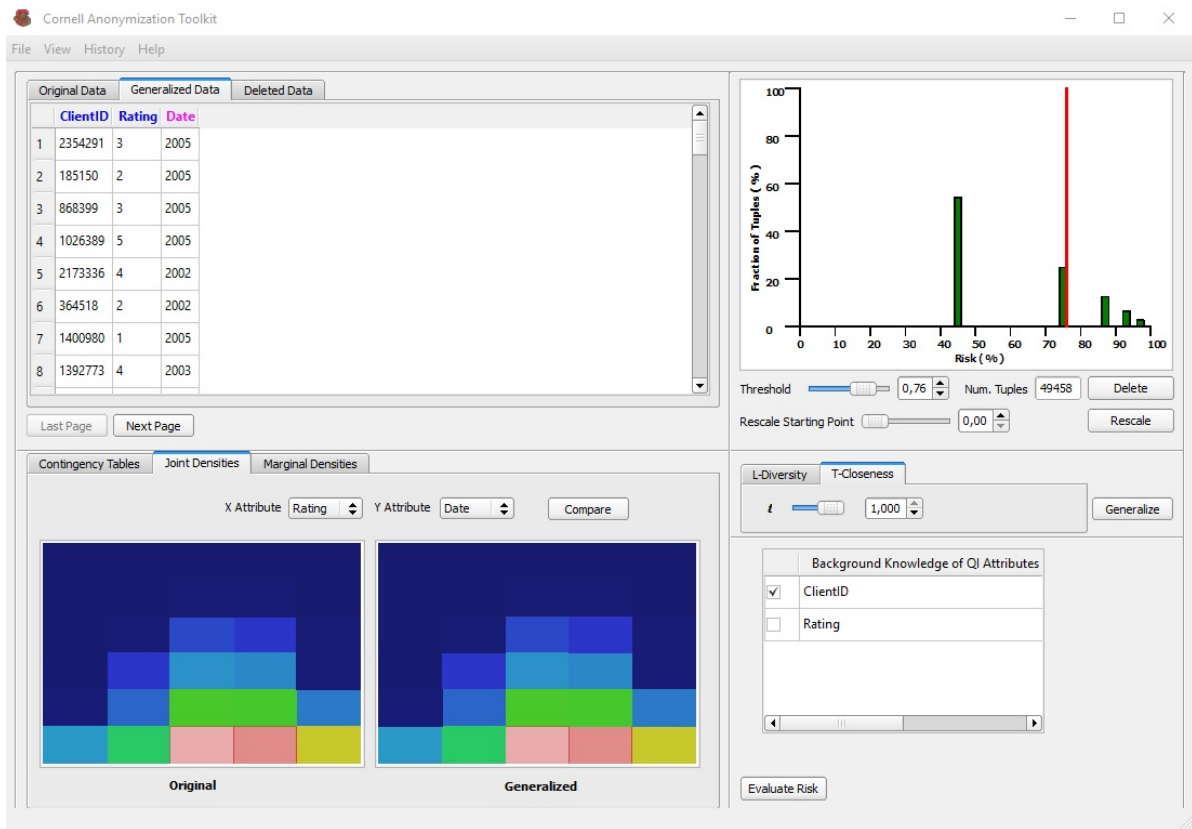


Figure 3: CAT Interface



According to the authors, CAT was designed with two purposes in mind. First, the software objective is to help the controllers and processors to correctly *anonymize* and analyse the risks of private information, in order to disclose secure data and protect the individuals contained in that same data. Second, CAT will allow the users to fully manipulate the *anonymization* process, giving them the power to adapt the configurations and analyse the quality of the *anonymized dataset* in a manner that satisfies them better [118]. Resuming, the software does what it is intended to do, but the biggest problem is pre-*anonymization*. Normalizing the data could take too many time on an analyst, beyond that, creating hierarchies will also be necessary, absorbing even more time and work.

- **Anonymization ToolBox** - Developed by the UT Dallas, this toolbox assembles diverse *anonymization* techniques for public use. After having some memory issues, caused by large *datasets*, the architecture was modified in order to simplify the *anonymization* of large *datasets*, the toolbox operates through an embedded database (*SQLite*) to reduce memory usage. Actually, the *ToolBox* only supports unstructured text files, but soon, the developers plan to add support for database connectivity, such as, Extensible Markup Language (XML) files and possibly the Resource Description Framework (RDF) data model. Currently, there are six distinct *anonymization* mechanisms over three particular privacy solutions, but in the future, developers hope to add more in order to have a variety of configuration options. While the program is reading the input data into the embedded database, all irrelevant attributes are deleted to improve *de-identification* efficiency, this way the *anonymization* is performed separately from the upload process [88].

This software accepts *.csv* files, but adding a dot in the final of every line of the *dataset* is required, in addition to that, a configuration file has to be set up with the *anonymization* preferences. In this file, the user will define each attribute type and the hierarchies intervals. This program runs in a command-line interface, is all write in *Java*, so a Java Development Kit (JDK) version must be installed, also uses *SQLite* in order to access the database. A *SQLite* package was included in the software, but the version is dated, so the

user needs to update this library in order to make it work. *Anonymization Toolbox* gives the chance to choose between four *anonymization* methods ("Datafly", "Mondrian", "Incognito" and "Anatomy"), some of these only works with a specified type of algorithm, while "Incognito" supports all of them. The method is defined in a configuration file, but doing it is not very clear, even with the documentation and a configuration file example provided, users could find difficulties to perform this task.

Finally, the *anonymization* process can go for few seconds to several hours, depending on the method chosen and the *dataset* size. In a test performed with *k-anonymity* and *Mondrian* technique, forty nine hours were spent only to build the hierarchy tree, and approximately four hours more to read and write the data. This application don't have any kind of risk analysis, so is difficult to rate the *anonymization* process effectiveness. After interacting with this software, we felt that was a good application, but for micro or small databases, in addition to that, sometimes the output is very confusing due to the appearance of random characters, requiring a normalization in order to be analysed more efficiently.

- ***sdcMicro*** - The anonymity must be respected by any *dataset* containing personal information, this can be achieved by applying SDC methods that will ensure a lower risk on data disclosure. The *sdcMicro* serves as an implementation of SDC methods to classify and *anonymize* micro-*datasets*, also incorporates all famous disclosure risk and perturbation schemes. The software executes automatic predictions of frequency counts, individual and global risk estimations, and at each *anonymization* step, presents statistical information about the amount of data lost and the utility of the *dataset* [89]. The optimization of processes was a concern when operating with bigger *datasets*, so they were highly optimized in terms of computational requirements. *sdcMicro* implements popular statistical disclosure methods for risk assessment, also, various perturbation methodologies were integrated, such as, shuffling, micro-aggregation, adding correlated noise, post-randomization, local suppression and others more [90]. With *sdcMicro* package, SDC methods can be applied in an easy, interactive and researching way [89].

### 3.3.2. Private Solutions

- **Privacy Analytics Eclipse** - This *de-identification* solution gives companies a safe way to share PHI for secondary use, reducing significantly the risk of *re-identification* and keeping the information utility intact. Always moving inside *Safe Harbor*'s limits, *Privacy Analytics Eclipse* allows to keep data quality and preserves compliance with many data privacy regulations. This software has the most sophisticated *de-identification* techniques, developed by privacy specialists. It was also built to embrace HIPAA's *Expert Determination Method*, that classifies the data attributes. He evaluates privacy risks and implement the right level of *de-identification* for the type of *dataset* submitted, offering an excellent balance between usefulness and privacy preservation.

The tool provides capacities to unlock PHI for secondary purposes, applying a risk-based technique, data collectors ensures that personal information is protection while keeping the *dataset* rich and real. *Privacy Analytics Eclipse* rewards data analysis professionals with a fast and very accurate *anonymizer* tool, always ensuring compliance with HIPAA and other legal regulations [91].

- **TIAMAT** - This tool is a bit different from all the other mentioned previously, that is because he doesn't *de-identify datasets*, instead he evaluates and compares *anonymization* algorithms, offers to data collectors the opportunity to check the precision and overhead of actual *anonymization* techniques. It executes real-time and interactive *anonymization* methods comparison, as well as QID modifications (*anonymizing* or removing) effects on the final product. Other features include, attribute statistics compilation, diverse information loss metrics and compatibility with commercial database engines. *TIAMAT* focuses on the *k-anonymity* algorithm, also other privacy-preserving paradigms that use QID generalization can be integrated, such as *l-diversity* and *t-closeness*.

The tool allows data publishers to analyse the accuracy and runtime performance of various *k-anonymity* techniques, finding also suitable parameters for *anonymization* [92], providing information that could be used to modify the *k-anonymity* algorithm.

- **SECRET**A - This tool also evaluates and compares *anonymization* algorithms for relational and transactional *datasets*. The system contains nine recognized algorithms assembled on a benchmark-oriented framework, allowing data collectors to request and analyse the performance of these algorithms.

*SECRET*A has two operating modes, "Evaluation" and "Comparison". The "Evaluation" mode can be used to set and appraise the efficiency of a certain algorithm, concerning to data utility and protection, as well as its effectiveness. The "Comparison" mode gives to data publishers the capacity to create and process benchmarks for multiple *anonymization* algorithms comparison. A graphical report of the analysis is presented, allowing fast and intuitive interpretation on the efficiency and execution time of different algorithms.

*SECRET*A is the only that offers full assessment and comparison of *anonymization* methods, "the Cornell *Anonymization Toolkit* demonstrates a single algorithm for relational data, also supported by *SECRET*A, while *TIAMAT* does not support algorithms for transaction data, neither methods for *anonymizing* relational and transactional *datasets*" [93].

### 3.3.3. Tools Comparison

This procedure will have in account two major factors (execution time and *re-identification* risk), these characteristic seemed the more suitable for the type of analysis performed here. To begin, we used three algorithms (*k-anonymity*, *l-diversity* and *t-closeness*), applied to the solutions under analysis, switching various parameters in order to obtain the best results. The tests were performed using the *Netflix dataset*, we have chosen it because a *re-identification* attack was successfully applied on it, later, this technique will be replied on a *anonymized dataset*, testing the security of the software chosen to be a reference to the platform on creation. The *dataset* chosen has three attributes (*ClientID*, *Rate*, *Date*), and contains approximately 233.000 records.

Results will be presented in the table format to give a better understanding, many of them are not here due to a selection of the best results, e.g., *ARX* includes various utility measures that can be combined, the choice was "loss" as measure and "rank" as aggregate function, picking this

combination was based on the authors guide [84] and the purpose of this *anonymization*. In the first two columns of the tables, is represented which attributes are defined as sensitive or QIDs, a few combinations of attribute classifications were tested, but the two presented in the tables were considered the most suitable and secure. In the *re-identification* risk section, a column called "All" represents the three attributes combined risk of identifying a client, and in this case, is the most important due to the *datasets* anatomy (all the attributes are essential).

The ideal coding model will be considered 50% *suppression* and 50% *generalization*, this because the first hides values completely (if *suppression* is at 100%) and the second replaces values for something less specific (normally around the mean), so that is the recommended option in order to keep the information utility.

- ***k-anonymity***

In this privacy model we used two values for "k" (2 and 1000), normally, they are the maximum and minimum values allowed by the software tested, also an intermediate value (100) was tried but the differences were small, so we discarded him due to the little importance he had. In terms of execution time, *ARX* is much better than *Anonymization Toolbox*, this advantage is around a magnitude of a hundred times. The risk analysis shows (table 1) that two *anonymization* were better (yellow and green), but one has lower risk than the other.

Settings					Execution Time (sec.)			De-identification Risk (%)			
SA	QID	Type (k)	Supression	Generalization	ARX	CAT	AT	ARX			
								ClientID	Rating	Date	All
-	clientID, rating and date	2	100%	0%	0,132	-	-	0	0	0	0
-	clientID, rating and date	2	50%	50%	0,528	-	116	68,89	74,93	62,89	97,16
-	clientID, rating and date	2	0%	100%	0,115	-	-	74,79	74,93	0	93,68
-	clientID, rating and date	1000	100%	0%	0,425	-	-	50	74,31	60,49	95,12
-	clientID, rating and date	1000	50%	50%	0,425	-	115	50	74,31	60,49	95,12
-	clientID, rating and date	1000	0%	100%	0,184	-	-	68,89	74,93	0	92,2
clientID	rating and date	2	100%	0%	0,018	-	-	0	74,88	99,87	99,98
clientID	rating and date	2	50%	50%	0,134	-	91	0	74,88	99,87	99,98
clientID	rating and date	2	0%	100%	0,016	-	-	0	74,93	0	74,93
clientID	rating and date	1000	100%	0%	0,017	-	-	0	0	0	0
clientID	rating and date	1000	50%	50%	0,018	-	102	0	74,63	62	90,67
clientID	rating and date	1000	0%	100%	0,017	-	-	0	74,93	0	74,93

Table 1: *k-anonymity* Performances

On the green one, the attribute "clientID" has 0% of *re-identification* risk against 50% on the yellow, besides, on the "All" column the advantage goes again to green with 90,67% in opposition to 95,12%.

Looking at these results, is conclusive that the *anonymization* highlighted in green is the better one. As predicted,  $k=1000$  is more efficient than  $k=2$ , that is because he introduces more entropy on the *anonymized dataset*. On the green *anonymization*, attributes classification is better because the "clientID" is defined as identifiable against a QID on yellow, also, the difference of times is considerable here, and possibly even more expressive on big *datasets*. However, analysts on data protection should have into account that *k-anonymity* is not a good method to *anonymize* high-dimensional *datasets* [94].

- ***I-diversity***

Equally to the previous model, we used equal values for "l" (2 and 1000), for "c", the value (0,001) was fixed due to the authors guide recommendation for this type of *dataset*. On ARX, four other variants (distinct, Shannon-entropy, Grassberger-entropy and recursive) of the algorithm were tested, the one that presented the best result in this case was the recursive variant. At execution time, ARX leads one more time, with an advantage of around a minute (table 2).

Settings					Execution Time (sec.)			De-identification Risk (%)			
SA	QID	Type (c,l)	Supression	Generalization	ARX	CAT	AT	ARX			
								ClientID	Rating	Date	All
clientID	date	(0.001, 2)	100%	0%	2,903	-	-	100	72,73	0	100
clientID	date	(0.001, 2)	50%	50%	3,072	-	112	100	74,93	99,87	100
clientID	date	(0.001, 2)	0%	100%	2,903	-	-	100	72,78	0	100
clientID	date	(0.001, 1000)	100%	0%	0,141	-	-	0	0	0	0
clientID	date	(0.001, 1000)	50%	50%	3,072	-	111	100	74,93	99,87	100
clientID	date	(0.001, 1000)	0%	100%	0,101	-	-	100	74,93	0	100
clientID	rating and date	(0.001, 2)	100%	0%	0,58	-	-	100	74,63	62	100
clientID	rating and date	(0.001, 2)	50%	50%	0,58	-	125	100	74,63	62	100
clientID	rating and date	(0.001, 2)	0%	100%	0,554	-	-	100	74,93	0	100
clientID	rating and date	(0.001, 1000)	100%	0%	0,054	-	-	0	0	0	0
clientID	rating and date	(0.001, 1000)	50%	50%	0,558	-	133	100	70,31	60,49	100
clientID	rating and date	(0.001, 1000)	0%	100%	0,501	-	-	100	74,93	0	100

Table 2: *I-diversity* Performances

Again, is highlighted the two better risk analysis, on green the one with the best result and in yellow the second one. On both *anonymizations* selected, the attribute "ClientID" and the "All" column had 100% risk of *re-identification*, so differences were made by the other two attributes. On the green, "rate" has 70,31% against 74,63% on yellow, as the "date" attribute, percentages goes 60,49% for green and 62% on yellow. Here, the differences are small on the two selected results for the *ARX anonymizations*, however, the other solution loses again in terms of time and output presentation. Although these tiny variations, the green highlight *anonymization* is better than the others, in both execution time and *re-identification* risk. The "winning" attribute classification is the same as before, where "ClientID" is considered as a SA and the other two attributes as QIDs. Furthermore, this "I" value guarantees a well balanced dispersion of SAs through the *dataset*.

- ***t-closeness***

In this model, we used the values 0,001 and 1 for "t" variable, which are the minimum and maximum values accepted. On *ARX*, there are three distinct distance measures (equal, hierarchical and ordered), the one that presented better results was ordered distance. Surprisingly, CAT had the best execution time, while *ARX* was in average fifteen seconds slower (table 3).

Settings					Execution Time (sec.)			De-identification Risk (%)							
SA	QID	Type (t)	Supression	Generalization	ARX	CAT	AT	ARX				CAT			
								ClientID	Rating	Date	All	ClientID	Rating	Date	All
clientID	date	0,001	100%	0%	1,011	-	-	0	0	0	0	-	-	-	-
clientID	date	0,001	50%	50%	1,737	-	107	100	74,93	0	100	-	-	-	-
clientID	date	0,001	0%	100%	1,737	0,2	-	100	74,93	0	100	0	0	0	0
clientID	date	1	100%	0%	16,125	-	-	100	74,93	99,89	100	-	-	-	-
clientID	date	1	50%	50%	16,125	-	106	100	74,93	99,89	100	-	-	-	-
clientID	date	1	0%	100%	16,125	0,4	-	100	74,93	99,89	100	94	11	0	100
clientID	rating and date	0,001	100%	0%	2,178	-	-	0	0	0	0	-	-	-	-
clientID	rating and date	0,001	50%	50%	2,789	-	96	100	0	0	100	-	-	-	-
clientID	rating and date	0,001	0%	100%	2,464	0,2	-	100	49,96	0	100	0	0	0	0
clientID	rating and date	1	100%	0%	38,392	-	-	100	74,93	99,89	100	-	-	-	-
clientID	rating and date	1	50%	50%	41,025	-	92	100	74,93	99,89	100	-	-	-	-
clientID	rating and date	1	0%	100%	41,558	0,7	-	100	74,93	99,89	100	95	7	0	100

Table 3: *t-closeness* Performances



In this case, CAT also produced better results in terms of *re-identification* risks, so, on green is highlighted the two *anonymization* performed by this application, both using 0,001 as "t" value, but with different attributes classification. On yellow, is possible to see ARX best result, but he loses on both parameters, execution time and risk *re-identification*. To be noted, the displayed risk analysis are given by the own software, so there is no way (that we know) to ensure that this is an impartial or accurate analysis, this point is made due to a little scepticism in relation to CAT risk analysis results being less detailed than ARX. Instead of the previous privacy models tested, CAT wins with great margins on both evaluated criteria.

The results presented are not very precise and detailed when compared to ARX, and 0% of *re-identification* risk arises a little suspicion even with generalization. Still, CAT shows the percentage of risk for every tuple, also, the output is very organized and delivers the type of *anonymization* expected.

### 3.3.4. Final Results

All the tests were performed on a machine with the following specifications: Intel Core i5 2.5 GHz, 6GB DDR3, HD 750GB 5400 rpm and Windows 10 x64. On testing these applications, we noted they all read different types of input, some read Comma-Separated Values (CSV) files, others support *.xls* files, some only imports micro and meta data and others only supports *.data* files. Another problem was the setup of hierarchies, while ARX creates them almost automatically, in CAT they need to be created all manually and that could take several hours to do. In some cases the normalization can be very hard, adding the hierarchical organization and the *anonymization* configuration, turns the pre-*anonymization* process quite painful and slow.

After analysing the three most popular privacy models separately, in two of them, ARX was more efficient than the other two applications. This software was not only better on *anonymization*, but also on the user experience aspect, it is very detailed software, with a lot of information on the most probable *re-identifying* attributes, also, with different views of data before/after the *anonymization* process. ARX has a tab ("Explore results"), where the user can see different solution spaces that can be applied to the *dataset*, providing a better and more suitable *anonymization*.



In terms of configuring, a huge variety of features that allows the user to build a configuration suitable to his intents, giving also a prediction on the best *anonymization* for a certain *dataset*. On every screen, *ARX* displays a help button that presents detailed information, this provides a kind of guide to the most inexperienced users. Resuming, the best software was *ARX*, very competent on his purpose and incredibly complete in terms of configuration options, his simple but intuitive interface helps the user to understand what has to be done and how to do it.

An aspect noted, the privacy model with the best results on the risk analysis was *k-anonymity*. That happened due to the nature of the *dataset*, not too big and with few attributes, when a lot of entropy is introduced the *re-identification* is almost impossible. But, as mentioned before, this algorithm has some problems, and that is why *l-diversity* and *t-closeness* were created. Actually, the last one is more used due to less *re-identification* issues, some analysts combines privacy and coding models to increase security.

# Development

## 4.1. *Web Anonymizer*

After researching about what is *de-identifying* a *dataset*, how to protect PHI and analysing the available *de-identification* software, the time to developing the *Web Anonymizer* had arrived. Before starting creating the application, an architecture was designed to offer some guidance on constructing the *RESTful API*. Besides that, other elements were created to give directions on where to go and what to do next, a conceptual model was built to help constructing transitions between resources, also, a *mockup* was designed to represent an early model on the interface visualization expectation.

After concluding the preparation phase, the next step was starting to build the application itself. Below, we will present all the steps made through that process, starting with the planing phase, then the technologies used to make this all project work, focusing later on the *ARX API*. Finally, an explanation of the *RESTful API* implemented and how she manages communications between client and server, concluding with a presentation on the results obtained from the tests executed.

### 4.1.1. Architecture

In order to give the platform some structure, a layered architecture was created, grouping the different components in sections. This layered separation is often used on *web* platforms, on this particular architecture is separated in three layers (tiers). Each of these sections is responsible for a particular and unique processing, they communicate with each other through different frameworks and languages, so, aggregating all those tiers in a single framework simplifies hugely this interconnection process (figure 4). The layers create were called "Presentation Tier", "Business Tier", and "Data Tier". This division is very usual on *web* application, after an analysis, it seemed appropriate for this project too. The utility of these layers will be now explained.

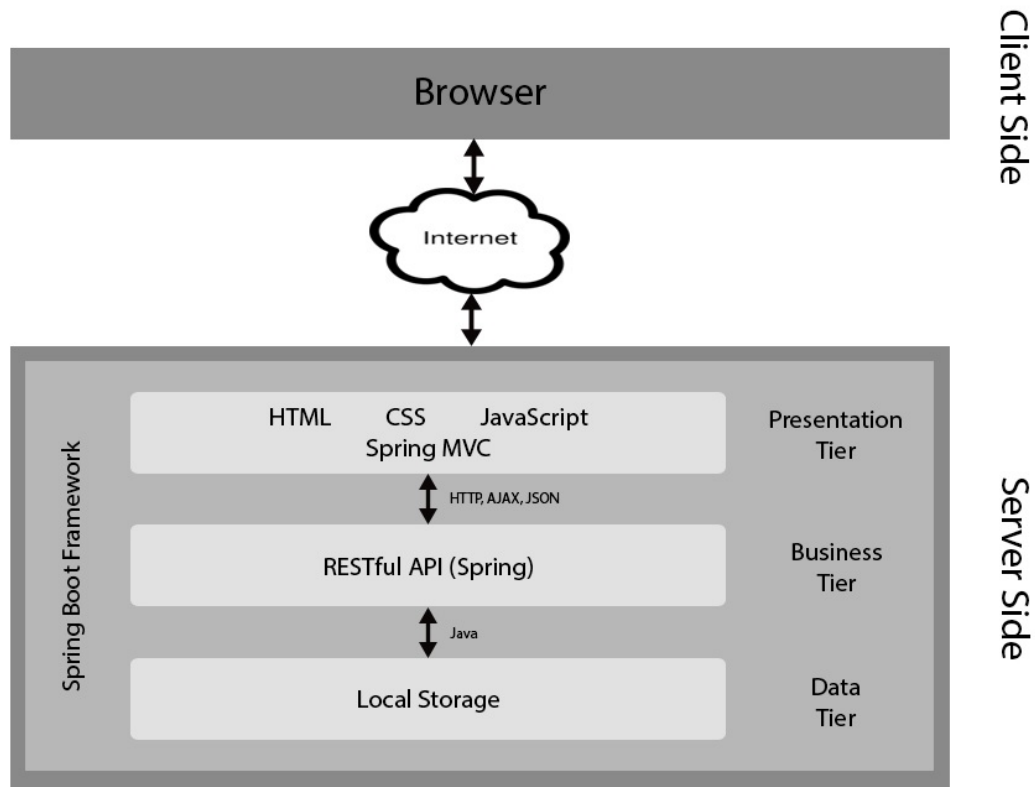


Figure 4: *Web Anonymizer Architecture*

- Presentation Tier** - This layer contains the user oriented functionalities, they are responsible for managing user interaction with the system, and generally consists of components that provide a common bridge to the business layer core. Presentation tier components allow users to interact with the application, but they don't process data or directly access databases neither other storage media, also, it generally provides a bridge to the business tier. Usually, is here that the data is validated, before being submitted to the business tier and stored in the database. So, the main responsibility of this layer is translating tasks from the user interface to server commands and vice-versa.

- **Business Tier** - This layer implements the core functionality of the system and encapsulates the relevant business logic. It generally consists of elements that execute rules and separates them from the user interface and data access. These components are responsible for: ensuring referential integrity by maintaining relations and record sets for a transaction or task, communicating with components in the presentation tier using disconnected record, guaranteeing a clear separation between presentation and application logic, and finally, is also responsible for processing and moving data between the two surrounding layers.
- **Data Tier** - The data tier provides access to the databases and storage devices used by a three-tier applications. It is responsible for retrieving data and transforming into a suitable format for the rest of the application. Essentially, this layer stores and retrieves information to the business tier for processing and eventually to the presentation tier. Initially, the architecture design included a database to save the original *datasets* and the resulting *anonymizations*, posteriorly, on adapting the *ARX API* we noted that there was no need to use a database in this version. As a proof of concept, saving the files locally was sufficient enough to perform the *anonymization* tests required to back up the work done.

#### 4.1.2. Interface

The interface of any application needs to be intuitive enough, in a way that the user understand what he can or cannot do, all that without any internal or external help. Note that most failures of human-machine systems are due to poor interface designs that don't acknowledge people's capacities and weaknesses. There are some recommendations to obtain an excellent human-computer interaction, such as, is important to know the characteristics of the people that will use the platform, what the application will do in general and prioritize the various objectives, and finally, the technologies available and their limitations as well as possible alternatives to them [116].

Having a functional and attractive design is also important to captivate and visually help users operating the application, there is not a right way to do it, so various interpretations on

"Design Concept" can be found in the related literature [117]. To create the interface of this platform we took into account four popular concepts:

- **Affordance** - The perceived and actual fundamental properties of the object that determine how could possibly be used, providing strong clues to the operations of things, e.g., buttons are made to be pressed.
- **Mapping** - The set of possible/natural relations between objects, links what can be done and what is perceived possible. Mapping is an important concept in the design and layout of controls. Summarizing, is the relationship between moving a control and watch the result in real world.
- **Feedback** - Represents visual effects exhibited only after an action is made, i.e., sends information back to the user about what action has actually been done and what result was accomplished.
- **Visibility** - One of the most important aspects of design, an interface must have visible features, inferring the right messages to the user. Elements must be well defined and detached, labels clearly highlighted, avoiding always overlapping elements.

Before building the interface is important to create a conceptual model, this defines how you want the user to see the system. A conceptual model is an explanation, usually highly simplified, of how a system works, it doesn't have to be complete or very accurate. These models provide predictive and explanatory images for a better understanding of the interactions. He can be presented as a storyboard or paper prototype, illustrating and documenting an intended or observed model, also useful for design, communication and analysis.

Normally, it doesn't specify any activities ordering, representing instead, the transitions (actions taken) between those activities [116] [118]. In the model created for this platform (figure 5), is possible to see the different menus and their transitions, in the top part of the image, they all belong to the configuration process, the user can go back and forward through these menus without changing the values inserted.

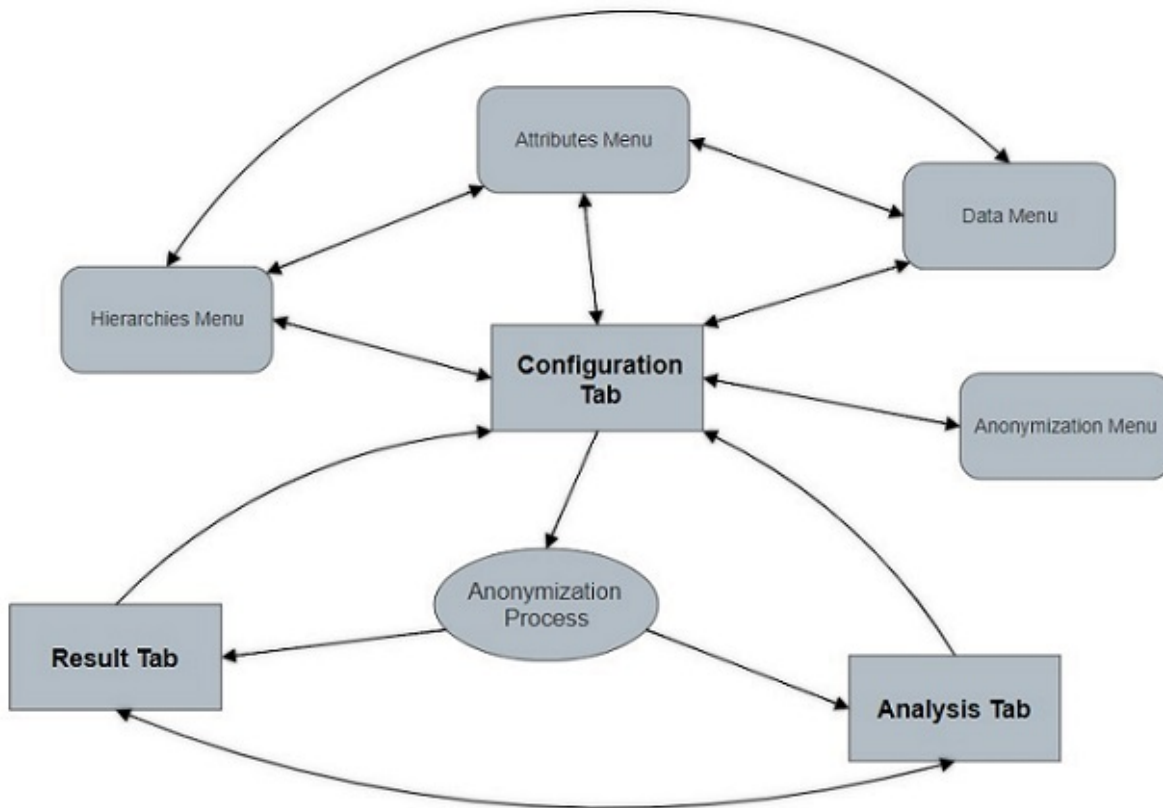


Figure 5: *Web Anonymizer* Conceptual Model

When the user believes that he has the right settings, the *anonymization* process can be initialized, after that, "Result" and "Analysis" tabs are automatically filled with the results and statics of the last *anonymization* process executed. Moreover, the user can always go back to the "Configuration" tab and change it, after, he can start a new *anonymization* process again, the corresponding results and statics will also be properly loaded.

Before designing the interface, a *mockup* was drawn (figure 6), representing the various screens that the user will see on the application, these images served as a guide for constructing the layout, there were not an exact replica of what will be seen in the final product. In the final version of the application some changes were made, these happened to achieving a better design, or because they were more suited to the functionalities needed.

Configuration Results Risk Analysis

Data Attributes Hierarchies

Input Data:

	ClientID	Rating	Date
1	173827	2	2004-02-01
2	464627	3	2005-10-05
3	324523	5	2003-06-23
4	987365	4	2001-09-11
5	263627	5	2002-03-15
6	72910	4	2003-07-26
7	346736	4	2005-08-30
8	64774	3	2005-09-04
9	63612	5	2001-12-25
10	87923	2	2004-11-08
11	54637	2	2003-04-18
12	23693	3	2002-05-29

Privacy Models

☒ k-Anonymity k =

☐ l-Diversity l =  c =

☐ t-Closeness t =

Codification Technique

Supression  Generalization

Configuration Results Risk Analysis

Data Attributes Hierarchies

clientID  Rating  Date

Insensitive

Privacy Models

☐ k-Anonymity k =

☒ l-Diversity l =  c =

☐ t-Closeness t =

Codification Technique

Supression  Generalization

Configuration Results Risk Analysis

Data Attributes Hierarchies

clientID

Bottom Value:  Top Value:  Interval:

Privacy Models

☐ k-Anonymity k =

☐ l-Diversity l =  c =

☒ t-Closeness t =

Method

Supression  Generalization

Configuration Results Risk Analysis

Original Data

	clientID	Rating	Date
1	2354291	3	2005-03-24
2	185150	2	2005-06-27
3	868399	3	2005-12-16
4	1026389	5	2005-04-07
5	2173336	4	2002-01-14
6	364518	2	2002-05-22
7	1400980	1	2005-05-06
8	1392773	4	2003-11-30
9	2473170	4	2005-09-18
10	1625755	4	2005-02-25
11	1744873	3	2005-06-09
12	53679	4	2005-06-13
13	712664	3	2001-07-23
14	1990901	4	2001-04-03
15	2308631	4	2005-03-12
16	479062	5	2005-11-12

Anonymized Data

	clientID	Rating	Date
1	[5176,10352]	3	2005
2	[165632, 331264]	2	2005
3	[20704, 41408]	3	2005
4	[662528, 1325056]	5	2005
5	[82816, 165632]	4	2002
6	[662528, 1325056]	2	2002
7	[20704, 41408]	1	2005
8	[0, 2588]	4	2003
9	[2588, 5176]	4	2005
10	[82816, 165632]	4	2005
11	[662528, 1325056]	3	2005
12	[82816, 165632]	4	2005
13	[165632, 331264]	3	2001
14	[20704, 41408]	4	2001
15	[5176,10352]	4	2005
16	[662528, 1325056]	5	2005

Configuration Results Risk Analysis

Choose Attribute(s):

☒ clientID

☒ Rating

☒ Date

Attribute	Fraction of Tuples (%)
Rating	74.628455%
Date	99.86811%
Client ID	100%
Rating, Date	99.96783%
Client ID, Date	100%
Client ID, Rating	100%
Client ID, Rating, Date	100%

Fraction of Tuples (%)

Risk (%)

45

With the *mockup* and conceptual model ready, the next step was creating the interface itself. Some modifications were introduced in relation to the *mockup* due to design issues, also some difficulties in adapting certain elements with the functionalities required were founded. The conceptual model was also very useful, mostly to understand the transitions between views, moreover, it affected the final design with the necessity of allowing the user going back and forward through all the screens. A big concern creating this layout, was displaying the elements over the screen, giving enough space for future elements to be inserted. Finally, another aspect we had attention was the fact that the tables presentation varies on size. These tables suffer modifications on their dimensions according to the *dataset* uploaded characteristics (number of attributes, records length and the number of rows), so the tables were positioned and configured in a way that they can expand without overlapping other elements (figure 7).

The screenshot displays the 'Web Anonymizer' application interface. It features a top navigation bar with 'Configuration', 'Results', and 'Risk Analysis' tabs. Below this, there are sub-tabs for 'Data', 'Attributes', and 'Hierarchies'. The 'Data' tab is active, showing a table of data with columns: ID, Sexo, Idade, Seguro, SUS, CF, and Murmur. The table contains 17 rows of data. To the right of the table, there is a 'Privacy Model' section with three radio buttons: 'k-anonymity', 'l-diversity' (which is selected), and 't-closeness'. Next to these are input fields for 'k =', 'l = 2', and 'c = 0.001'. Below the privacy model is a 'Codification Technique' section with a slider ranging from 'Generalization' to 'Suppression', with a value of '100' indicated. At the bottom right, there is a large button labeled 'Anonymize!!!'. At the bottom of the interface, there are navigation buttons: 'Prev.', '1', '2', '3', '4', '5', '62', and 'Next'.

ID	Sexo	Idade	Seguro	SUS	CF	Murmur
1	M	55	false	199-84-4481	92	false
2	M	31	true	140-27-2094	73	true
3	F	4	true	143-43-6413	104	false
4	F	56	false	897-58-6307	107	false
5	F	38	true	558-90-8282	86	true
6	F	2	false	698-91-9658	46	false
7	F	68	false	536-02-9292	115	false
8	F	88	true	122-84-3836	126	true
9	F	58	false	277-76-7208	141	true
10	F	1	true	227-12-0060	58	false
11	M	86	false	888-48-6197	48	true
12	F	57	true	585-84-1268	148	false
13	F	91	true	877-42-1002	98	true
14	F	17	true	723-73-1439	146	true
15	F	62	true	424-42-9253	48	true
16	M	98	true	374-15-6140	31	false
17	F	54	false	383-39-3351	157	false

Figure 7: *Web Anonymizer* Interface

Being this a prototype, the design was not a major concern, so we chose to keep it simple and more important functional. In the future, a professional designer should review the layout, implementing some improvements in order to turn the interface visually more attractive.



### 4.1.3. Technologies

#### 4.1.3.1. *Client-side*

In the application *front-end*, we used four technologies: HyperText Markup Language (HTML), Cascading Style Sheets (CSS), *JavaScript* and Asynchronous JavaScript and XML (*AJAX*). The *web* page presented on the user's browser was built on HTML, created from scratch, exhibiting a simple but intuitive interface that guide the user through a configuration process that sometimes could be a complex task. In order to turn the HTML more attractive, we applied CSS, giving a better visual presentation to the interface. As seen on the "Interface" section, some changes were made in relation to the original design, improving the graphical visualization in terms of interactivity, even if that is not the primordial objective of this work, a little bit of styling is always eye catching.

In this platform is also used *JavaScript*, in both synchronous and asynchronous tasks. Also, the platform needed a technology to send and receive data from a server asynchronously (in the background), all this without interfering with the display and behaviour of the existing page. After a little research, *AJAX* seemed the obvious choice to perform this type of actions.

Now, a more detailed analysis will be made on each of these technologies, why they were chosen to those specific functions and how the platform took advantage of them.

- **HTML** - An *web App* is basically an HTML application, this language describes and defines the content of a *webpage*, to complement is normally used CSS for design purposes and *JavaScript* to functionality. Commonly, HTML is used to build the content of *web* application, here, we used the most recent version (*HTML5*). This HTML version includes a set of new features, such as, new types of tags, a greater standardization and consequently a cleaner code. Furthermore, form validation is now HTML native, removing the need of *JavaScript*, therefore, server processing time is lower making the *web* pages load faster.

Maybe the biggest problem in creating a *web* platform using HTML, is making it work smoothly on the different browsers, so, a lot of multiple browser testing is required. The ap-

plication created was mostly tested on *Chrome* browser, making it work in various browser was not a major concern, the point here was to prove that works over the Internet independently which browser is used. In the future, if the platform is published, it will be important to be fully compatible with the major browsers.

- **CSS** - Is a highly effective complement to HTML, providing an easy control over the layout and presentation of website pages by separating content from design. This way, is possible to create different design types for each page. CSS works by creating rules, they can be applied to multiple elements within the site, also helps reducing repetitive tags from the HTML code. Furthermore, the browsers download the CSS file once (when the page is loaded for the first time) and caches it before loading the website pages, this makes the website load faster enhancing the overall user experience.
- **JavaScript** - A dynamic and object-based programming language, normally used to build interactive *web* pages. The majority of websites supports it without the need for *plug-ins*. Initially, was *client-side* focused, but his powerful performances made the programmers start using also on *server-side*, being actually not far behind compared to other *server-side* technologies. Finally, as a language that works on both the *front-end* and *back-end*, allows code re-utilization and facilitates data transfers, this way, developing applications is much faster than using different technologies on both sides.

In this project, *JavaScript* was used in two different ways, inserted on HTML code and through external script files. The code written in the own HTML ("index.html" file) is used to make diverse asynchronous tasks, but also on requests made to the server, mostly preparing the data to be sent or organizing the received one. External files containing *JavaScript* are defined in the "index.html" header with the tag "<script>", followed by the source path file. The option of using external files was motivated by the integration of open source files, these *JavaScript* files resolved some needs the project had, such as, manipulating the menu tabs or showing the value in codification bar. *JavaScript* was very helpful in terms of *client-side* interaction, at the *back-end* there was no need to use *JavaScript*.

- **AJAX** - This technology allows to update a *web* page without reloading it, also permits to request and receive data from a server after the page is loaded, also allows sending data to the server in the background. These characteristics were needed in this project, e.g., when a *dataset anonymization* is performed, the interface has to present the results to the user without reloading the page. This way, the configuration values were kept intact, allowing the user to correct them as much as he wants, until a desired result is achieved. This technology has an important role in *web* applications, is through her that the HTTP requests are performed, so every time the clients need a resource from the server an AJAX call is made (figure 8).

```
1  ...
2  $.ajax({
3      url: "/uploadFile",
4      type: "POST",
5      data: new FormData($("#upload-file-form")[0]),
6      enctype: 'multipart/form-data',
7      processData: false,
8      contentType: false,
9      cache: false,
10     success: function () {
11         // Handle upload success
12         $("#upload-file-message").text("File succesfully uploaded");
13     },
14     error: function () {
15         // Handle upload error
16         $("#upload-file-message").text("File not uploaded (perhaps it's too much big)");
17     }
18 });
19 ...
```

Figure 8: Upload AJAX Code

For example, in the *Web Anonymizer*, a *dataset* is uploaded to the server for an eventual *anonymization*, this has to happen without reloading the page, so, AJAX had to be used. Another technology was used in order to send and receive data from the server, is called JavaScript Object Notation (JSON), and is useful for parsing *JavaScript* objects into text. This is needed when the client wants to exchange data with the server, because this

data has to be in text format. JSON converts *JavaScripts* objects in text, and vice-versa, the data is saved as pairs of key/value and they are kept as an array of *strings*. This technology was used in all the AJAX requests created, parsing the data sent to the server, or making the reverse process for the received array of data.

#### 4.1.3.2. *Server-side*

- **Spring Boot Framework** - *Spring Boot* is a very popular *Java* based framework for building applications. It is basically a suite, pre-configured set of frameworks/technologies to provide an easy way to have an *RESTful web* application up and running with almost zero configurations [119]. Unlike many other frameworks which focuses on only one area, *Spring* framework provides a wide verity of features addressing the modern business needs.

This framework gives flexibility in configuration (XML, *Annotations* and *JavaConfig*), however, with the increase of features, the complexity also grows and configuring *Spring* applications could become an exhausting and error-prone task. This choice was made due to certain advantages, such as, embedded *Tomcat* that quickly get up a running a server, without code generation neither requirement for XML configuration. Another advantage is the use of "pom.xml" file that simplifies a lot the *Maven* configuration, this file contains information (dependencies and *plugins*) about the project that are used by *Maven* to build the project.

The *Spring* framework provides extensive support for databases, working with Structured Query Language (SQL), and using HTTP and *Hibernate* to access and manage them. *Spring Data* also provides an additional level of functionality, due to the repository implementations, that accesses directly on interfaces and using conventions to generate queries from method names. It is often convenient to develop applications using an in-memory embedded database, obviously, this will not provide persistent storage. *Spring Boot* can also auto-configure embedded *Derby* databases, simply including a build dependency to the embedded database [120].

After analysing other available *RESTful* frameworks, the choice was *Spring*, mostly due to the compatibility with Representational State Transfer (REST), that manages the application from user interface to data storage. Another reason was the semi-automatic configuration that easily starts a server, also, because there is a lot of *Spring Boot* documentation and forum discussions that provides a lot of help.

- **REST** - Before the 90's, REST, was a term rarely used. Most of the time misused and generally misunderstood, even if HTTP and HTML standards (REST base), were widely developed and popular. However, in 2000, the term REST started to be used more often, a huge contribution to that was a thesis paper by *Roy Fielding*, that provided guidance on how to apply those concepts in applications [105]. Nowadays, when speaking in *web* platforms is common to call them *RESTful*, even if sometimes they don't respect some REST practices.

A big issue verified on nowadays APIs, is that once they published, they don't change anymore. When an application is *RESTful*, the architecture must be adapted to changes, this is crucial due to the incessant adjustments that the *World Wide Web* suffers everyday. In small systems not prepared for modifications is easy to apply changes, but when the API grows is more difficult to do so, especially if the system is not *RESTful*.

Besides the scalability problem, there are two mistakes that the developers make commonly, duplication of effort and avoidance of hypermedia [106]. Every developer creates a different API, and most of the time they are a collection of pieces from various others APIs, i.e., is created an API with elements that were not designed for that purpose. Reusing code can work, but has to be analysed very carefully before applying, standardization could also help stopping this waste of effort. The other problem is related to the misunderstand of hypermedia (links), consequently, developers often underestimate the power of it, turning their API unable to handling changes gracefully.

The *web* today is based on three technologies, the Uniform Resource Locator (URL) naming convention, the HTTP protocol and the HTML document format. The first two are relatively easy to work with, but HTML is a bit more complicated, mainly due to the diverse

data formats that compete to take HTML place. An ideal API should implement the same principals that make the *web* so easy to use, i.e., the user needs to figure out for himself how to use it without too much help or information.

A *web* application without resources don't serve for much, they are usually something that can be stored on a computer, e.g., an electronic document, a *dataset* or the result of executing an algorithm. These resources must have an URL, this way the servers and clients can trade them unmistakably due to their unique address [105]. When a GET request is made, the server uses a representation that captures the current state of the resource required, this contains information (size, encoding or data) about that resource depending on what was requested.

Although, there are restrictions on what the client can do with a resource. In *RESTful* systems, clients and servers communicate through messages that follow a predefined protocol, called HTTP. It defines eight different kinds of HTTP request, the most common are the following:

- **GET** - Request the representation of a resource, identified by an URL, is just a request for information. Sending a *GET* to the server don't have any effect on the resource itself, and usually the response code for this request is 200 (OK).
- **POST** - Create a new resource based on the given representation. When the *POST* request is sent, a representation of the resource to be created is embedded in the request's entity-body. Commonly, the response code for this type of request is 201 (Created), giving the information that the resource was created. A location parameter, inside the request's header, contains the URL to this new resource. Another frequent response code is 202 (Accepted), which means that the new resource is not yet created, but he will be eventually.
- **PUT** - Replace the state of a resource with the one defined in the given representation. An usual sequence of requests is, a *GET* to obtain the resource representation, followed by a *PUT* with the representation received inside the payload of that request.

- **DELETE** - This type of request is normally sent when the client wants a resource to be eliminated, however, the server can refuse this order. If a *DELETE* request succeeds, the server can respond with a status code 200 (OK) or a 202 code (Accepted). If a *GET* request is sent to a resource that has already been eliminated, the server will reply with an error code that could be a 404 (Not Found) or 410 (Gone).

The URL naming convention, is another concern in the REST universe. This is just the address of a resource, which a client can use to get a representation, technically, says nothing about the resource or its representation. Eventually, a HTTP request is sent to the server with a URL, then sooner or later, an exchange of requests will happen modifying the resource state.

This changes exemplifies the capacity of connecting the resources to each other (hypermedia), this technique can solve or at least mitigate usability and stability issues found in nowadays APIs [107]. Unfortunately, this strategy is not a standard, even though is implemented in several technologies. Hypermedia consists on the server informing which HTTP requests the client can make in the future, the server knows what could happen, but is the client that decides what will really occur. The truth is that this technique is still a bit confusing and controversial, making it rarely used. But when applied, results most of the times in APIs with poor capacities of managing changes, consequently they produce very inflexible *web* applications [106].

The performance is also a subject approached by the *RESTful* ideology. Clients can make requests as much as they want, however, some of those are useless and they need to be treated that way. The HTTP protocol presents several solutions for pointless request, such as, caching, conditional requests and compression. Authentication is also a concern here and should be implemented in *web* applications, is recommended to use a mechanism that hides user credentials on HTTP request exchanges [105], e.g., symmetrical or asymmetrical key cryptography is a popular solution. REST also worries with managing errors, ideally, libraries deals with both network and server errors, and is very important to

always give feedback to the user. Finally, documenting the API is describing a *web* service in human readable text, this can be generated by some tools or created by the application developers. This document provides useful information for developers, helping them to understand the "semantics" of the resources, media types, link relations, and so on. A detailed *RESTful* documentation should [107] approach the following subjects:

- All resources and methods supported for each resource
- Media types and representation formats for resources in requests
- Each link relation used, the HTTP method and the resource identified by the link
- All fixed URLs that are not supplied via links
- Query parameters used for all fixed URLs
- URL templates and token substitution rules
- Authentication and security credentials for accessing resources

Creating a *RESTful API* can improve an application in diverse aspects, like, performance, scalability, flexibility, visibility or portability. Furthermore, following this guideline will also turn an application more stable, interactive and faster, however, if a developer violates one of the REST principals, these advantages will not be guaranteed, and his API will not be strictly *RESTful*. Every time the server is started, *Spring* executes a complying REST platform, of course, the API as to be previously programmed to be *RESTfull*.

#### **4.1.4. ARX API**

Implementing this API, has revealed to be an amazing choice due a simple and organized code formulation, beyond that, a well written public documentation, plus a deductive online *javadoc* with all packages and classes included in the API. The aim of the programming interface is to provide *de-identification* methods to other software systems, being the library provided by *ARX*, often simpler than interaction with the graphical tool [84].



The developers had divided this software into six sets of packages, each corresponding with a step in the *anonymization* process and risk analysis. These packages are classified as, data import and specification, hierarchy generation, privacy criteria, measuring data utility, data analysis and representing the solution space. In each of the packages exists a set of classes that performs actions related to a purpose (normally associated to the package name), e.g., the class *Data.java* (included in the *org.deidentifier.arx* package) implements a generic representation of the input data.

The principal classes, the ones that are always called in the *anonymization* process, they are: *ARXConfiguration*, *ARXAnonymizer* and *ARXResult* (figure 9).

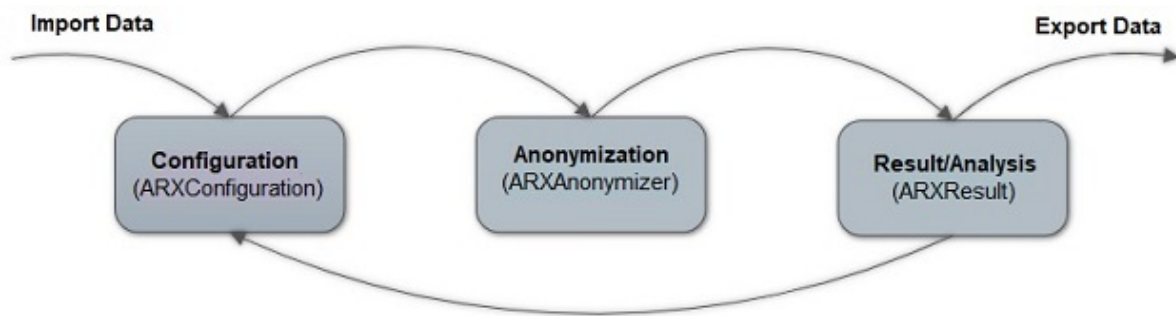


Figure 9: *Anonymization Process*

The *ARXConfiguration* defines a group of settings (privacy model, codification value, ...) that are later sent to the main class (*ARXAnonymizer*), this configuration allows multiple parameters, offering the user an opportunity to create suitable *de-identified datasets* [85]. *ARXAnonymizer* offers several methods to define parameters and execute the ARX algorithm, according with the configuration given by *ARXConfiguration*, the outcome of this process will produce a result of the type *ARXResult*.

The *ARXResult* receives that result and presents it to the user, the class also includes a series of methods that gives useful information, such as, execution time or information loss. In order to present a risk analysis, *ARX API* takes the result and apply various methods, offering a

diversified set of risk information, all the classes that performs risk analysis can be found inside the "Risk" package.

Other classes are also important in the *anonymization* process, e.g., the *DataType* and *AttributeType* provides data and attribute types as static variables respectively. Another example, is the *DataHandle* class, allowing users to interact with the data, by performing operations such as sorting, swapping rows or reading cell values. Handles treat input and output data, they can be used for research subsets of such data. Finally, *DataSource* class provides configuration options for importing *datasets*, encapsulating all information required to access a database as well as the data properties, the last step is loading the *dataset* in memory, that is achieved by creating an instance of the class *Data*.

In terms of scalability, increasing the size of a *dataset* and the number of QIDs, will consequently expand the solution space. These changes, provokes the creation of an additional column, due to the multiplicity of the solution space by the number of levels in the associated hierarchy. To solve this scalability problem, and the need to compare execution times from different *datasets* scales, the authors defined a normalized execution time. This results from the product of an arithmetic division of the execution time by the number of transformations on the solution space, all this divided by the number of QIDs presented in the *dataset* [84]. However, if a privacy model that doesn't require frequency distributions (e.g., *k-anonymity*) is used, *ARX* performance is much better, fundamentally in terms of execution time. Also, the space-time trade-off implemented, allows a memory consumption reduction [121]. To test scalability, the *ARX API* developers used a *dataset* containing information about approximately 1.2 million individuals, and that is already considered as a large *dataset*.

#### **4.1.5. Web Anonymizer Overview**

In this section, an overview of the platform created will be presented, the different aspects of this application will be described. A demonstration of what was used from the original API, but also what was created in order to make this project work and the reasons to make it this way, all that will be explained in the following section.

#### 4.1.5.1. Adaptating the ARX API

In order to integrate the *ARX API* in the platform, a few changes had to be made, some classes have been modified and others were created. The original API was very efficient, so in terms of the *de-identification* process little was modified, however, the need for a *web* platform forced the creation of some classes and methods in order to connect the two tiers (business and presentation). Firstly, will see a bit more on the classes modified, and next we will show all the classes created to turn this platform accessible over the Internet.

##### Classes Modified

The classes that needed to be modified were "ARXConfiguration" and "ARXMain". In the first one, all variables that contain configuration values inserted on the platform by the user were being included, also we needed to add the related *getters* and *setters*. The reason for this adjustment is related with the *controller* created, when its called, he sends the parameter values to the API using the *getters* defined in this class. The "ARXMain" was a base class for the *anonymization* process classes (also known as *anonymizers*), we found it in the *ARX API* examples folder, created by the authors of this software as a demonstration of *ARX* capabilities.

- **ARXConfiguration** - This is a generic configuration for "ARXAnonymizer", all the possible options for the *anonymization* process can be found here, but nothing was modified. Nonetheless, these preferences were not all used in the current version of the platform, but thinking in scalability, all configurations were left available for a future upgrade. As mentioned before, due to the need of "mainController" in using *getters* to obtain the parameter values, we added all the variables used to keep the users configurations (figure 10), also, the corresponding *getters* and *setters*.

The variable "anonymType" saves the type of privacy model chosen, the "anonymValue" keeps the value of "k" or "l" depending on the model selected, "cValue" stores the "c" value for the *l-diversity* algorithm, the "tValue" saves the value of "t" for the *t-closeness* method, "codiValue" keeps the percentage of *generalization/suppression* chosen as cod-

ification technique, the variables "atribType" stores the type of each attribute, finally, the remaining ones saves the defining intervals values used to create the hierarchies for the different attributes.

```
1
2 public class ARXConfiguration implements Serializable, Cloneable {
3
4     //Variables used by the controller to pass the anonymization configurations to the ARXAnonymizer
5     String anonymType;
6     int anonymValue;
7     double cValue;
8     double tValue;
9     int codiValue;
10    String atrib1Type;
11    String atrib2Type;
12    String atrib3Type;
13    long atrib1Top;
14    long atrib1Bot;
15    long atrib1Int;
16    long atrib2Top;
17    long atrib2Bot;
18    long atrib2Int;
19    long atrib3Top;
20    long atrib3Bot;
21    long atrib3Int;
22
23    //Getters and setters
24    public String getAnonymType() {
25        return anonymType;
26    }
27    public void setAnonymType(String anonymType) {
28        this.anonymType = anonymType;
29    }
30    ...
}
```

Figure 10: "ARXConfiguration" Code Sample

These were the only changes performed in this class, we only had to find a correct way to send the configurations to the *controller*, these informations subsequently will be forwarded to the "ARXAnonymizer" class, then the *anonymization* process will be triggered using the configurations received.

- **ARXMain** - This class represents a sort of template for the *anonymizers*, made available by the authors as an example of a possible implementation. The use of this model simplified the API integration, cause many of the existing methods were useful for the platform being

built, such as, the data handler or the return of statistical results. Besides containing useful methods, this class was also kept untouched due to its strategical position, i.e., adding methods to this class will be available to any type of *controller* configuration a developer creates, in this platform only one *controller* ("mainController") was implemented. Right below, in the "mainController" topic, we will explain in more detail what is a *controller* and his role in *web* application.

The utility of this class is based on the possibility of adding methods that can be used in more than one *controller*, so, preserving "ARXMain" will help expand the platform in future versions, e.g, if someone wants to implement all the features that *ARX* presents, they will need to add the various methods in this class.

### Classes Created

Some classes had been designed to make the *API anonymize datasets* with the parameters chosen, furthermore, a *controller* was built in order to connect *ARX API* to the *web* server. An explanation of the methods implemented in those classes will be given, also, a thorough description on the *controller* due to the importance of this component in the project.

- **ARXMainAnonymizer** - The purpose of this class is "simply" taking the configurations set by the user, and transform them in "ARXAnonymizer" configurations, then the *anonymization* process is executed, lastly, the *anonymized dataset* is returned as "result". There are some declaration that needs to be highlighted for a better understand on their relevance. For example, in line 3 (figure 11) is perceptible that a new "DataSource" will be loaded, calling a method named "createCSVSource", i.e., the "source" will save the "test.csv" as input file, type of encoding, separator character and a boolean that defines if the *dataset* contains a header. On line 6, the *anonymizer* will define the data to be *anonymized*, the CSV file is saved on the "source" variable. The lines 9 and 12 represent the declaration of news instances on the corresponding classes, the first one creates a new *anonymizer* with default configurations, while the second declaration creates a new configuration without tu-

ple suppression (if no codification value is set). In line 17, the *anonymization* process is executed, saving the outcome as a "ARXResult".

This class supports various methods and handlers, allowing the presentation of different statics and other informations about the *anonymized* data. Besides that, it is possible to see that the *anonymizer* created execute a method called "anonymize", this one performs a data *anonymization* using the data and configuration parameters set before.

```
1  ...
2  // Load the dataset file
3  DataSource source = DataSource.createCSVSource("original_datasets/test.csv", StandardCharsets.UTF_8, ',', true);
4
5  // Create data object with the original dataset
6  Data data = Data.create(source);
7
8  // Create an instance of the anonymizer
9  ARXAnonymizer anonymizer = new ARXAnonymizer();
10
11 // Creates a configuration object
12 ARXConfiguration config = ARXConfiguration.create();
13
14 ...
15
16 // Execute the anonymization algorithm
17 ARXResult result = anonymizer.anonymize(data, config);
18
19 return result;
20 ...
```

Figure 11: "ARXMainAnonymizer" Code Sample

All the methods used in this declaration were found in the API, they were quite easy to program them in order to define the *anonymization* process components, that was verified due the well elaborated documentation and coding examples made available by the *ARX API* creators.

- **mainController** - In the case of *RESTful web* services, controllers can help decrease the complexity of server/client communication, improving network efficiency, and letting servers implement complex operations atomically. They also let clients use the HTTP method *POST* to submit a request to trigger the operation. A controller is a component that can atomically make changes to resources.

In Spring, REST endpoints are just *Spring MVC controllers*. This class was defined as a *controller* using the annotation "`@controller`", is also possible to use "`@RestController`" (this combines "`@Controller`" and "`@ResponseBody`") annotation, to be noted that all these annotations result in *web* requests that returns data rather than a view [122]. The annotation used in this class was selected instead of "`@RestController`", because the various requests used here apply different annotations, and some are not supported by that annotation.

In one of these requests, the annotation "`@RequestParam`" is used, this bind a request parameter to a method parameter in the *controller*, i.e., informs the *Spring Boot* server that he needs to access the query parameter values from the request. Another annotation used for request is "`@RequestBody`", this binds a JSON value in the HTTP request body converting it into a *Java* object, in this case, a group of values is sent, and then, a "config" object that contains the configuration parameters for the *anonymizer* is created.

In all the requests made by the *controller*, two annotations are always there, the "`@RequestMapping`" and the "`@ResponseBody`". The first one, is used to map *web* requests into specific handler classes and/or handler methods, i.e., acts like a router, redirecting information according to the URL. If an HTTP request with the path "/" is made, the *Tomcat* server maps it to the home method, in this case is mapped to the "index.html". The "`@ResponseBody`" annotations indicates that the return value should be bound to the *web* response body (not placed in a *Model* or interpreted as a *View*), i.e., tells *Spring MVC* not to render a *Model* into a *View*, but instead to write the returned object into the response body, furthermore, the *Accept* header of the request will define that a JSON object should be returned [106].

The "mainController" implements five methods that performs requests, the first one is always executed when the server is started, and maps the "index.html" file as the home page. A second method, called "uploadFile", is responsible to upload the file selected by the user into the server's project folder, later, this *dataset* will be used to create an *anonymized* version. A method called "getRows", goes to the *dataset* passed as a parameter and copy 20 rows of that file, and sends them to the client as a list of strings into a JSON object.

Another parameter is also sent, called "position", this instructs the reader where to start copying rows. Every time the request is successful, the lines received are appended to the tables on the interface, and increments 20 units to the position variable. Another method created was the "countRows", when called this method goes to the file passed as a parameter and counts how many rows the file contains.

Lastly, a method called "anonym" (figure 12), execute the *anonymization* process with the configurations received, saving a copy of the *anonymized dataset* on the file path, and also, presenting on the platform interface a set of information defined on the "ARX-MainAnonymizer" class.

```

1  ...
2  @SuppressWarnings("deprecation")
3  @RequestMapping(value = "/anonym")
4  @ResponseBody
5  public String[] anonym(@RequestBody ARXConfiguration config) throws IOException {
6
7      // Array that saves the statistic information to be presented on the interface
8      String statics[] = new String [14];
9      // Array that keeps the re-identification risk per attribute
10     String attAnalysis[] = new String[7];
11
12     // Execute the anonymization process with the configurations received
13     ARXResult result = ARXMainAnonymization.anonymResult(config.getAnonymType(), config.getAnonymValue(),
14     config.getCValue(), config.getTValue(), config.getCodiValue(), config.getAtrib1Type(),
15     config.getAtrib2Type(), config.getAtrib3Type(), config.getAtrib1Top(), config.getAtrib1Bot(),
16     config.getAtrib1Int(), config.getAtrib2Top(), config.getAtrib2Bot(), config.getAtrib2Int(),
17     config.getAtrib3Top(), config.getAtrib3Bot(), config.getAtrib3Int());
18
19     // Save a copy of the dataset produced by the anonymization process
20     result.getOutput(false).save("anonym_datasets/test_anonymized.csv", ',');
21     ...

```

Figure 12: "mainController" Code Sample

#### 4.1.5.2. *RESTful API*

Before starting to build the *RESTful API*, some steps [107] were performed in order to organize the developing process, those steps were:

1. Making a list of all the resources that a client may request, not forgetting that this has to include the resources to send or receive from the server.



2. Drawing a state diagram for the API is always a good practice, this helps finding new elements on the creation process and will serve as a guide in the development. Each box on the diagram represents one kind of representation, therefore, a group of those will connect to the list made in the previous step. Normally, to connect representations is used arrows, they represent state transitions triggered by HTTP requests.

3. Choosing a media type, compatible with the defined protocol and application semantics.

4. Developing an HTTP server that implements the state diagram from step 2. A client that sends a certain HTTP request should trigger the appropriate state transition and get a certain representation in response.

5. Writing an API documentation, containing tutorials and client examples to help the users getting started, however, that's not part of the design and should be done after the application is fully operational.

It was seen before the importance of a *RESTfull API*, all communications to the server goes through her, so is important that the user or developers understand how this functions, thus, this section is dedicated to that. We will now navigate through the *Web Anonymizer*, demonstrating how this application works from the home page until the risk analysis, we will present images of the interface showing what happens when the user interacts with it, also, at each of these steps we will explain how the API executes that action.

In the home page (figure 13), there is button to upload a *dataset*, when the user presses that button, HTML opens automatically a search file window and then the user selects a file. This action will trigger a *JavaScript* "onchange" event, that calls a function named "uploadFile", inside this instruction block exists two AJAX requests (in the Appendix, an API documentation is presented with all the requests that can be made). The first one (called "/uploadFile") makes a *POST* request in order to save a copy in the server, if that request is successful, a "File successfully uploaded" message will appear on the interface, otherwise, an error message will warn the user that something went wrong.

The other AJAX request (called "/getRow") makes a "GET" request to the server asking for the first 20 rows of the file uploaded, then these lines are put in a table and presented to the

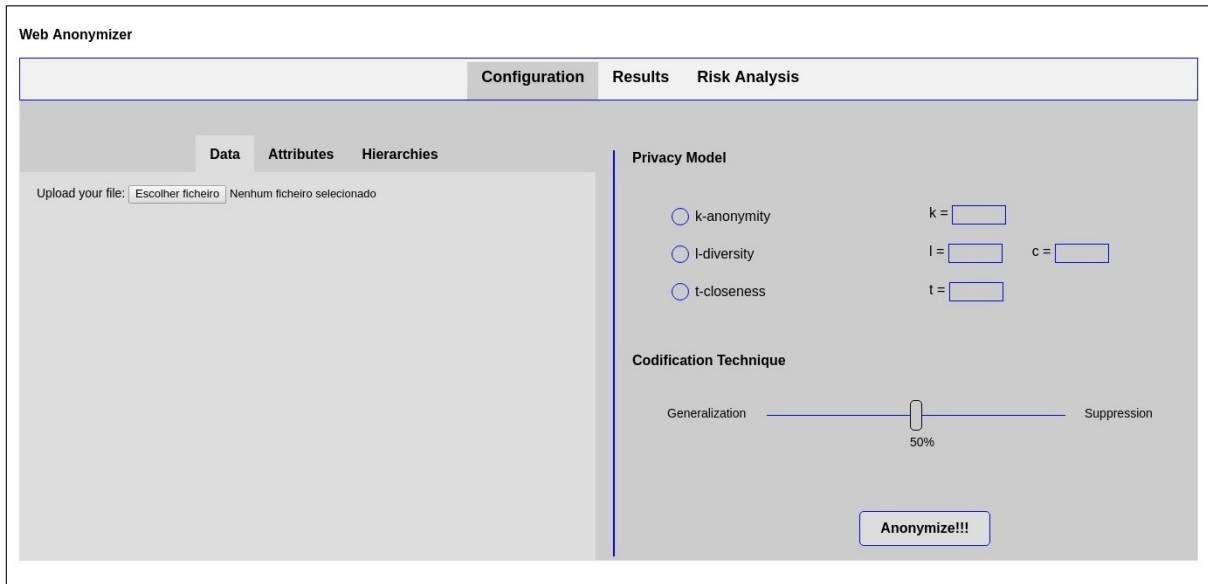


Figure 13: Home Page Interface

user on the interface (figure 14). The same process is made for a table in the "Results" tab, as both tables present the same data, we choose to do this at the same time, in order to reduce the number of server's request.

When the user makes a scroll on the table to watch the rows below, that action trigger a function called "scrollDataUpload", inside this block of code there is a "/getRows" AJAX request similar to previously showed. The difference lies in the necessity to append those lines to the already existing table without erasing the current ones, so the request asks for the next 20 rows in the *dataset*. When the data is completely loaded on the table, a message ("*Dataset* fully loaded") will appear instructing the user on that situation, also, the scroll bar will not go further down. To be noted, that this function only works for this table, cause one parameter of the AJAX request is the file name, so the other tables triggers different functions with the corresponding source files.

Below the table, a navigation bar is presented, when the table is loaded for the first time, the bar appears. To calculate the number of pages, a request ("/countRows") is made to the server asking for the number of rows contained in the *dataset*, after receiving the response, *JavaScript* make the calculations and present the bar with the total number of pages.

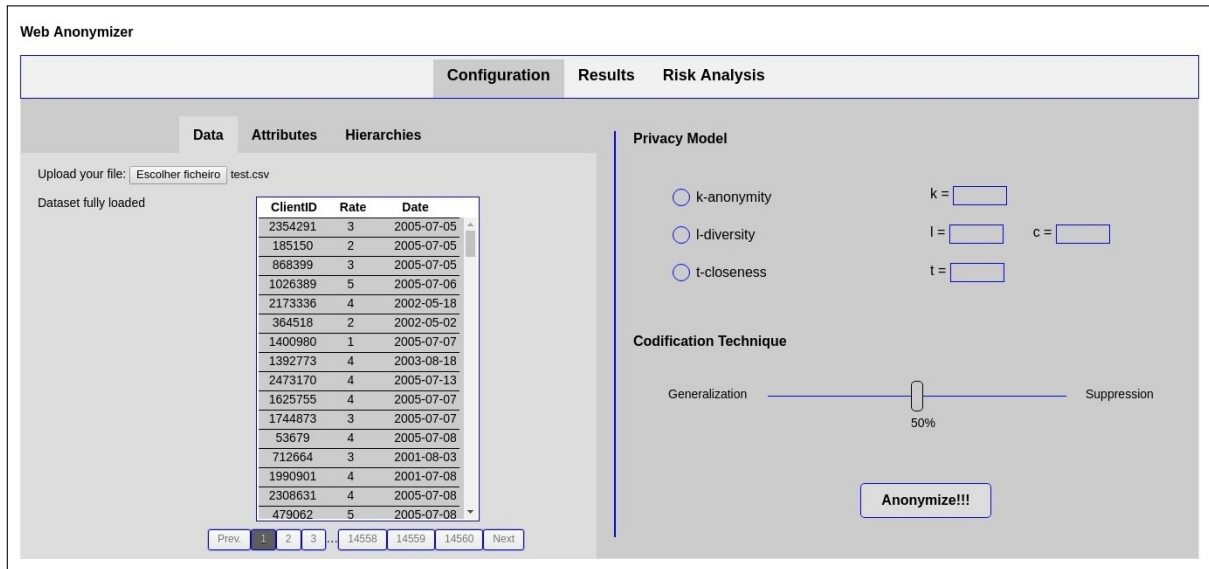


Figure 14: Uploaded *Dataset* Interface

After storing the uploaded file, the next step is configuring the *anonymization* process, the user can navigate through the configuration menu using the tabs on the left side of the screen.

As previously seen, the first tab is used to upload *datasets*, all these tab transitions are controlled by *JavaScript* code. The second tab leads to a configuration menu (figure 15) called "Attributes", as the name implies, here the user defines each attribute type depending on the level of privacy the user wants, but it also depends on the privacy model chosen.

In the third tab, the user defines hierarchies intervals (figure 16), again for each attribute, the top (higher) and the bottom (lowest) value needs to be set. The "interval" field receives the maximum number of elements in each interval, e.g., if the top value is 100, the bottom 0 and the interval is 20, the hierarchy for that attribute will have 5 levels.

Now, going to the left side of the configuration screen, we have two other settings to define, the privacy model and the codification technique. The user can select the model wanted using the radio buttons, also he has set the privacy model characteristic values. Also, a codification value as to be chosen, between full generalization or complete suppression of attributes, being still possible to define an intermediate amount of suppression, note that the range bar is related to

the percentage of suppression to be applied on the *dataset* records. After all configurations are set, and if the user agrees with those settings, he can press the "Anonymize" button, this will start the *anonymization* process.

Figure 16: Hierarchies Configuration View

When the button is pressed, an "onclick" event is triggered, this will call a function named "anonymize". Inside it, a number of variables were created to save the configuration set by the user. After that, they are converted into a JSON object using the method "JSON.stringify", this will create a group of key/value pairs. An AJAX "POST" request, called "/anonym", sends that JSON object to the server, there, the *anonymization* process is triggered. When this action is complete, the server saves a copy of the *anonymized dataset* in the "anonym\_datasets" folder, and sends back to the client an object containing various information related to the last *anonymization* process. Meanwhile, the interface changes automatically to the "Results" tab (figure 17), presenting to the user the original and *anonymized datasets*. The table that presents the *anonymized* data is filled before changing the tab view, this process is the same used on the other tables. An AJAX request is made, asking for the first 20 rows of the *anonymized dataset*, as explained before, this

is made by a similar request that only change the target file on the parameters send to the server. These two tables are completed using the same technique, i.e., when the user scrolls down the tables, a *GET* request is triggered. This event will append the received rows in the respective tables, this happens while the user scrolls down or until the file is not fully loaded into the table. This view allows the user to compare both *datasets*, this way he can analyse the levels of protection and utility applied, if the results don't match with the desired ones, he can go back to the configuration menu and change them in a more suitable way.

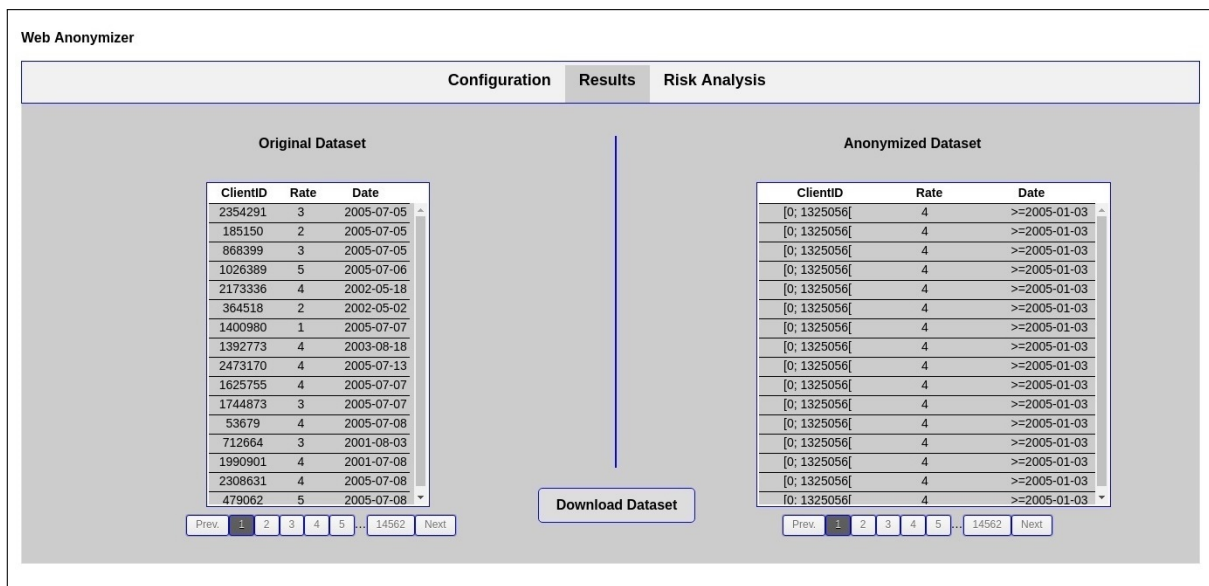


Figure 17: Results View

Changing to the "Risk Analysis" tab, the user will find some information about the last *anonymization* executed (figure 18), these shows details like the time spent or the level of information lost. An analysis on *re-identification* risk per attribute is presented, these values give an estimated percentage of an *re-identification* attack being successful only having access to one or more attributes data. Another risk analysis perspective is presented here, the *re-identification* risks for the prosecutor, journalist and marketer attacker models, these patterns were designed by the ARX creators.

In the prosecutor model, is assumed that the attackers knows that the *dataset* contains PII about someone he knows. The journalist model presumes that the attackers have zero knowledge on the *dataset*. In the marketer model, is assumed that the attackers don't have a specific target, they want to *re-identify* as many people as possible. However, successful *re-identification* attacks are low, because they are only considered when the majority of records are *re-identified* [84].

Web Anonymizer		
Configuration Results Risk Analysis		
Anonymization Statics	Re-identification Risk Per Attribute(s)	Attack Models Re-identification Risk
Time Spent: 21 segundos	Attribute(s): [Rate], Distinction: 0%, Separation: 75%	Prosecutor re-identification risk: 50%
Information Loss (Lowest): 0.001867401600000207	Attribute(s): [Date], Distinction: 1%, Separation: 100%	Journalist re-identification risk: 50%
Information Loss (Highest): 0.001867401600000207	Attribute(s): [Rate, Date], Distinction: 3%, Separation: 100%	Marketer re-identification risk: 3%
Suppressed Records: 435		
Download Dataset		

Figure 18: Risk Analysis Menu

In both, "Results" and "Risk Analysis" menu, a button was added at the bottom of the screen, this allows the user to download the last *anonymized dataset* produced. When the "Download Dataset" button is pressed, a AJAX request ("/downloadFile") is made to server, with the file name as a parameter, then the server respond with a *blob* (binary large object) type, containing the *anonymized dataset*, that is automatically downloaded to the user's computer. Finally, when the platform was created, we had into account that the user probably will make many *anonymizations* until he reaches the desired result, unless he is a professional and only one attempt is made.

As mentioned before, in the Appendix, there is an *RESTful API* documentation. A detailed explanation of each method used will be presented, what is his role, which arguments are sent,

what is received and how errors are dealt. This will help to understand the core of the platform, allowing users to work properly with it, or even helping developers to implement the API effectively.

#### 4.1.6. Project Folders Structure

The directory hierarchy in a project is essential in terms of organization, but more importantly as a functional structure.

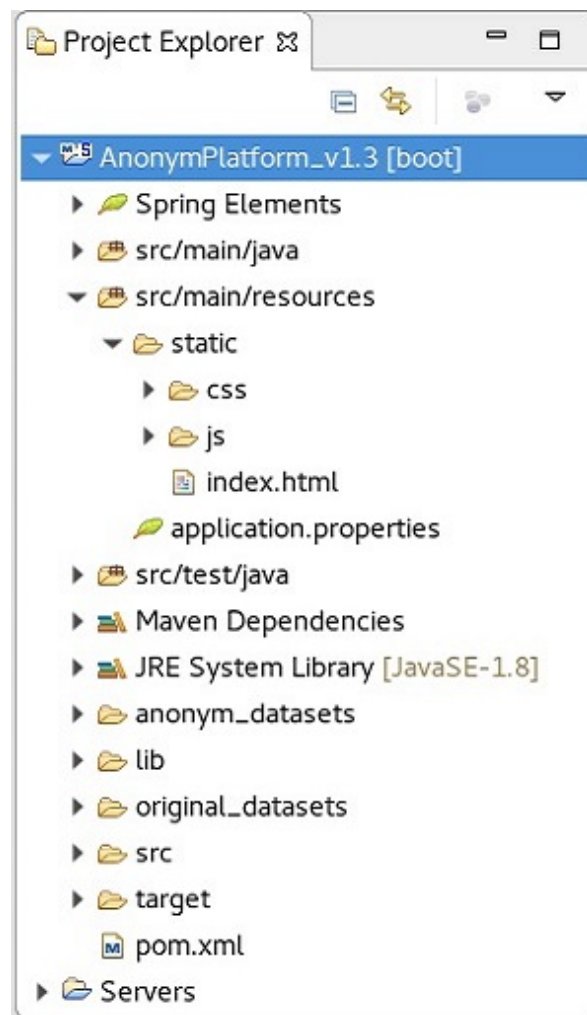


Figure 19: Project Folder Structure

Since this project uses *Spring Boot* framework, when a new *Maven* project is created, a default folder structure is automatically built, is recommended to respect it as much as possible to keep the project working properly. Little changes were made from the initial project structure (figure 19), those modifications were: the *ARX API* insertion in the "java" folder, as well as, all the other *anonymization* packages and classes created. All the *web* elements mention before (HTML, CSS and *JavaScript*) were put in the "resources" directory, the "lib" folder keeps the libraries (*JARs*) required by the API. Besides, two folders were created to store the *datasets*, one called "original\_datasets" to keep the *datasets* loaded into the server, and another called "anonym\_datasets" to store the *anonymized* data. This organization is crucial in projects with more than one developer, avoiding misunderstands and code duplication. Also, in terms of scalability, is important to keep a "clean" and well documented project, future developers will understand more easily what the code does and where to find it. Another important rule is the nomenclature of folders, packages, classes, and so on. The names given should always be related to the function, brief and use "CamelCase" when appropriate. Finally, is common to see comments and annotation in the code, that is essential to understand what a piece of code does, well documented projects ensure the future for themselves.

## 4.2. Tests and Results

After the *Web Anonymizer* was fully operational, a set of tests was performed in order to check not only if everything was really working properly, but also to evaluate the results produced by the platform. The tests executed here were evaluated using only one parameter, the execution time. In the "Risk Analysis as a Service" chapter, tests will be performed to evaluate two other important characteristics, information loss and *re-identification* risks.

To perform these tests, we adopted the *Netflix dataset* used previously, this will also give the possibility to compare performances between *ARX* software and this platform. The tests performed were similar to the ones done in the "Tool Comparison" section, with the same data and identical configurations.



In terms of functionality, everything the application presents in its interface is operational, i.e., all the configuration input elements receive correctly the values inserted, the *anonymization* process runs normally as well as the results presented to the user. Since this platform runs over the Internet, sometimes delays can be verified, e.g., when the *dataset* is loaded to the server, however, these delays are in the worst case a few seconds. Besides, after executing all the tests, nothing unexpected or irregular was verified in terms of functionality.

The server used to execute the test was allocated in the Faculdade de Ciências da Universidade do Porto as a virtual machine, he had the following specification: 4 cores 2,5 GHz, 8GB RAM and 50GB HD. The platform was placed in the server file system, to start it, the command "mvn spring-boot:run" had to be executed.

```
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Version             2.2.8
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://4ac7f745.ngrok.io -> localhost:8088
Forwarding           https://4ac7f745.ngrok.io -> localhost:8088

Connections          ttl    opn    rt1    rt5    p50    p90
                   6      0      0.04   0.02   60.32  61.27

HTTP Requests
-----
GET /favicon.ico           200 OK
GET /js/d3.min.js          200 OK
GET /js/jquery.csv-0.71.min.js 200 OK
GET /js/jquery-csv.js      200 OK
GET /js/jquery.min.js      200 OK
GET /js/template.js        200 OK
GET /css/style.css         200 OK
GET /                      200 OK

2017-08-28 15:48:00.625 INFO 25634 --- [main] s.b.c.e.t.TomcatEmbedd
edServletContainer : Tomcat started on port(s): 8088 (http)
2017-08-28 15:48:00.631 INFO 25634 --- [main] anonym.Application
: Started Application in 55.128 seconds (JVM running for 64.9
44)
```

Figure 20: Ngrok Tunnelling to Tomcat Server

After that, a software called "ngrok" was needed, this program creates a tunnel from any machine to the local server, this was also installed in the server's machine. To run this software, it was necessary to execute the command "ngrok http 8088" (8088 is the port chosen to run *Spring boot* server), after a while, the program gets *online* and returns a URL (figure 20), inserting it on the local machine browser will give access to the platform's interface. The *ngrok* also shows all the HTTP requests made to the server, in the image is possible to see the requests made for *JavaScript* and CSS files on loading the application page. This way, a real environment is simulated, and with it, all the constraints that could affect the results and functionalities. The most significant differences between *online* and local simulation were the upload and execution times, they were a little bit longer but no more than a few seconds.

#### **4.2.1. ARX vs. Web Anonymizer Performance**

The tests performed here are exactly the same with both platforms, i.e., we used the same *dataset*, as well as, the same configurations (tables 1, 2 and 3). In the graphic below, we present a comparison of the execution time on both applications, using different privacy models in order to have sufficient data for analysing the contrasts. As predicted, the risk analysis results were similar on both platforms. This happens because both use the same API, the modifications made did not have any influence on the *anonymization* process, they were applied at the handling level, i.e., the changes were related to the main objective of this work, and that was turning this API in a *web* application. The tests executed here used the better results obtained before we start implementing *ARX* software, these were applied in the *Web Anonymizer* and the results were a bit impressive. Comparing the execution times, *ARX* as a much better performance, an arithmetic mean was calculated to have a global idea in the differences. *ARX* had a mean execution time of 5,62 seconds, against the 49,08 seconds of the platform (figure 21), this shows that the *Web Anonymizer* is approximately nine times slower than *ARX*.

The principal reason for this to happen is because it function over the Internet, so the delays of sending and receiving the information to/from the server will always interfere, in this case there is not much to do, unless trying to optimize the code in order to reduce these delays.

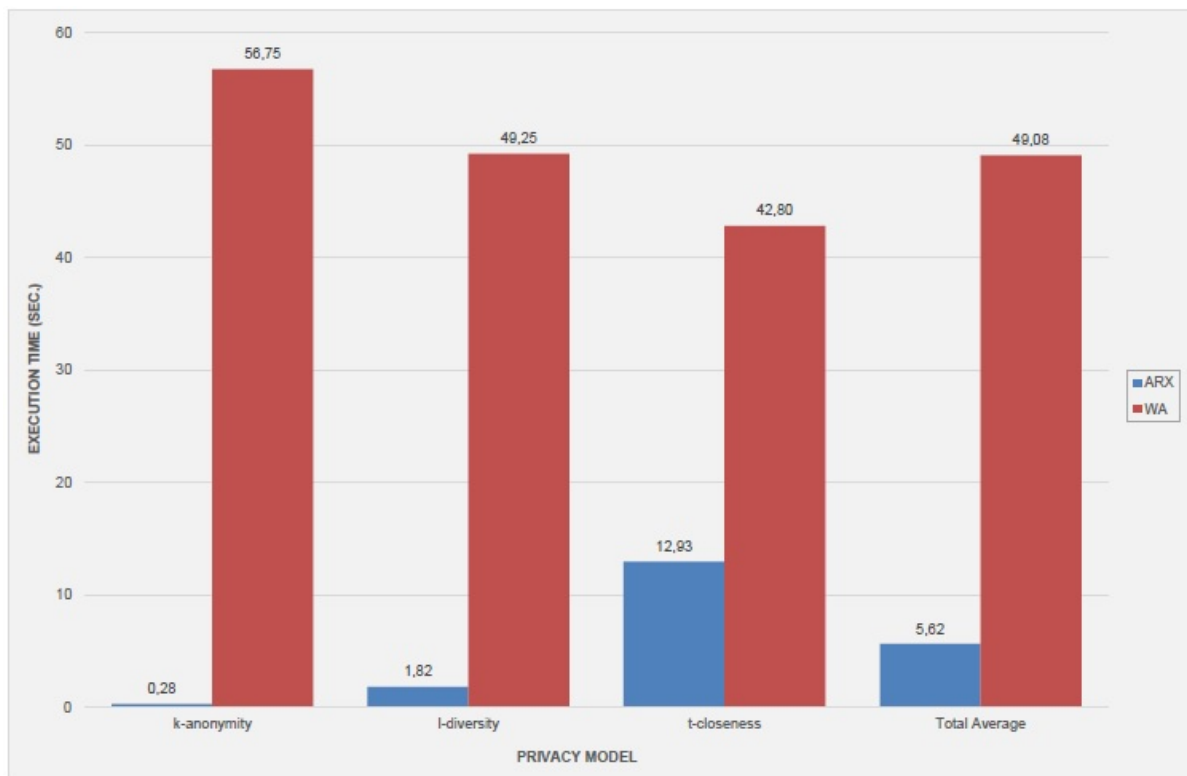


Figure 21: *ARX/Anonymization* Platform Performance Comparison

# Risk Analysis as a Service

Previously, we demonstrated the differences in terms of execution time between *ARX* and the application created, but we felt the need to demonstrate stronger evidences that the *web* platform created is efficient and reliable. So, we focused on the *datasets* produced robustness, i.e, we tested the effectiveness of *Web Anonymizer's* *de-identification* process in terms of privacy protection. Before building the application, we tested the *datasets anonymized* with the software chosen previously, an interesting method to do that was found. Reproducing the *Netflix re-identification* technique, initially performed by *Arvind Narayanan* and *Vitaly Shmatikov*, proved that the released database was vulnerable to privacy attacks [17]. In 2005, a group of academics from the University of *Aveiro*, used the same method to prove that privacy improvements can be achieved at a utility cost, compatible with tailored recommendation scenarios, using a simple partition-based *sanitization* method [109].

Initially, the objective of using this technique, with the help of those previous efforts, was proving that *ARX* could improve (or at least match) the level of privacy in the *Netflix* database published, guaranteeing at the same time its utility. Later, *anonymized datasets* produced by the *Web Anonymizer* were also tested using the same method, this way we obtained some comparative references between the two applications in terms of data protection. With this analysis, we wanted to prove that our prototype (*Web Anonymizer*) fulfils the objective initially proposed. That intent was, creating a *web* platform that *anonymizes datasets* based on user's configurations, presenting the result in a simple way and giving also the user a risk analysis on the *anonymized dataset*. Using this technique, not only allowed us to compare with the *ARX* risk analysis results, but also with the results achieved in the previously mentioned literature. We used this successful known case to legitimize this proof of concept, and even better, to validate all the work done in this project so far.

## 5.1. Purposes

Data is often released or sold to third parties allegedly in a *anonymized* form, but many times only some basic techniques of suppression or masquerading are applied on common identifiers, and without any type *re-identification* tests. These methods apparently protect information, but they are vulnerable to inference attacks, attributes like zip code, sex and birth date can be combined to identify a person. *Latanya Sweeney* demonstrated that with those three attributes, 87% of the US population could be identified [108].

As seen before, *Netflix* released a sanitized *dataset* containing film ratings for competitors to use, claiming that the client identifiers were suppressed, only appearing a random number as client identification. *Narayanan and Shmatikov*, crossed information with *IMDb* and *MovieLens* databases, *re-identifying* some clients from the *Netflix dataset*, uncovering even their apparent political preferences and other potentially sensitive information [17].

*Narayanan and Shmatikov* presented a way to identify some subscribers if a little knowledge about a client is acquired by the adversary, however, the client needs to be included in the *dataset*. To identify someone, the attacker needed to know that the dates had a maximum of fourteen days error, the rating knowledge didn't have to be accurate, and even some of the ratings and dates known could be completely wrong. The algorithms used were quite robust, and they proved that if a client record was identified on the *dataset*, probably it was not a false positive [109]

Data publishers have to decide the type of each attribute before disclosing data, which of them are available to the user and which needs to be protected. The *k-anonymity* algorithm guarantees that QID tuples occurs in at least *k* records in the *anonymized* database, but this don't provide any privacy, because the values of SAs associated with a given QID may not be sufficiently diverse, [99, 110] or the adversary may know more than just the QID. Furthermore, *k-anonymity* technique completely fails on high-dimensional *datasets* [94], such as, the *Netflix Prize dataset* and in most of real-world *datasets* with personal information.

In high-dimensional *datasets* there are many attributes, they are usually sparse (each record typically has a non-null value defined only for a small fraction of the attributes), also, the distri-

bution is typically long-tailed (there is a small number of attributes that have non-null values for many records, while there is a large number of attributes that only have non-null values for a few records). The problem with these big *datasets* is they are easily recognizable, and that means they are vulnerable even if the attribute values are suppressed or generalized. *Generalization* and *perturbation* techniques are usually applied on low-dimensional *datasets*, sometimes they are used in high-dimensional too, but that is not recommended at all.

As mentioned before, records on big *datasets* are very distinguishable, so *dataset* partitioning has been increasingly explored as an alternative technique for protecting privacy. One of the advantages of partitioning is that the resulting *dataset* values remain unchanged, what changes is the records they are associated with. To safeguard privacy in high-dimensional *datasets*, is used some forms of horizontal and vertical partitioning, *fragmentation* and *disassociation* [111] are often used to achieve protection on this type of *datasets*. However, they have little differences, while *disassociation* uses *k-anonymity* to guarantee privacy, *fragmentation* vertically divides a high-dimensional *dataset* into a set of low-dimensional *datasets*, they are called fragments, achieving an almost "perfect" privacy-utility trade-off.

Resuming, this attack was performed to prove that *re-identifying* persons on the *Netflix dataset* was possible, *Narayanan and Shmatikov* explored the *k-anonymity* failure in protecting high-dimensional *datasets*, also the sparsity and similarity presented in that *dataset* helped to identify *Netflix* clients. This paper [17], was also created to alert about some *anonymization* weaknesses, presenting some issues with a privacy model, sanitization methods or attributes type definition. Solutions for those problems were also presented in that paper, besides, many other publications appeared with possible answers to these and other problems related with *anonymization*, one of those cases was the University of Aveiro paper [109], largely used in the tests performed in this chapter.

## 5.2. Re-identification Test Procedure

*Narayanan and Shmatikov* concluded that as little as auxiliary information is, she is crucial for identifying a client record from the *Netflix Prize dataset*. They also revealed that with eight movie ratings (two of these could be completely wrong), and a fourteen days error, approximately 99% of records could be *re-identified* in the *dataset*. Furthermore, with two ratings and a three days error, the percentage could reach 68% of identified subscribers. The level of perturbation that must be applied to the data, in order to break the algorithm used, will definitely cripple the utility of the *dataset*. As seen before, they also agreed that the usual techniques used by *k-anonymity* algorithm, such as *generalization* and *suppression* [96] [99], do not ensure privacy protection, and in any case they fail completely on high-dimensional data. Moreover, the knowledge of non-null values columns, on most records, reveals as much information as knowing the specific values of those columns, therefore, such techniques don't increase security mainly because SAs are untouched [17].

To help preprocessing and organizing the data, two frameworks were used, the *Netflix Recommender Framework* and *Kadri Framework*. They were very popular during the contest period, and provided functions to efficiently process the *Netflix dataset* text files, as well as, implementing some recommendation algorithm primitives, namely, average, matrix factorization and prediction blending [113, 114]. These frameworks provide also a function that allows to clean the probe data from the *dataset*, removing that data of the training set increases the reliability of *RMSE* results [109].

A *Netflix Commons* library was created by the authors, in *Java*, to provide quick and memory-efficient access to the *dataset*. Heuristics were used to improve access times to data without requiring more memory for the representation, also, the *Scoreboard-RH* algorithm uses this library. The preprocessing step implies matrix factorization and attribute cardinality counting, in this case, is the number of ratings per user. It was developed in *C++* and uses the *Kadri Framework*. This process generates three files, the *Scoreboard-RH* algorithm uses these files and the original *dataset*. Therefore, the algorithm generates three other files, a fragmented version of the

*dataset*, a probe file in accordance to the resulting *dataset* and a pseudonym mapping file for evaluation purposes. In terms of performance, a safe cardinality value for movies was introduced in the implementation, if the film has a number of ratings above this value then is not considered to be a centroid, consequently, the number of movies sorted is reduced by cardinality, and therefore, the execution time.

The most important step was evaluating the *dataset* risk *re-identification*, to achieve that, the *Scoreboard-RH* algorithm was used. The low levels of similarity between clients in the *dataset*, served as a base for this algorithm. Besides, the small number of ratings on less popular movies makes this *dataset* very distinguishable, so, based on the size of films support set (*supp*) Narayanan and Shmatikov assigned different weights for each movie. They also defined a variable (*sim*) to save the coefficient of similarity between two rows. This algorithm defines a *Score* function that specifies a score for each record (*r*) in the *dataset*, this value is set by the resemblance between attacker's auxiliary information (*aux*) and the record itself.

$$Score(aux, r) = \sum_{i \in supp(aux)} wt(i) \times sim(aux_i, r_i)$$

where

$$wt(i) = \frac{1}{\log |supp(i)|}$$

When *Score* function is performed for all the rows, it is calculated the difference between the two highest *Scores* obtained, after, the result is divided by the standard deviation ( $\sigma$ ).



$$\frac{max - max_2}{\sigma} < \theta$$

If the resulting value is lower than a parameter called "eccentricity" ( $\theta$ ), then no match was found, otherwise, the record with the highest score is considered a successful match. However, the *Scoreboard-RH* algorithm is not 100% reliable, some failures were found when this algorithm is calculating the best *Score*. The *Scoreboard-RH* fails to find the correct record in two occasions, when a wrong record is defined as the highest *Score* and when the correct record has a *Score* lower than the second highest *Score*.

The tool used to measure utility, was the "F-measure" [112], this tool doesn't rely on the QID assumption, instead, he uses an approach based on the results of data mining workload to quantify utility. Measuring the utility was not really necessary here, but we still used this tool to compare the results obtained with those achieved by *Aveiro* academics.

The *datasets anonymized* by both applications were submitted to this *re-identification* process, this was a quite long procedure, due to the configurations that had to be made, also, a normalization of the *datasets* had to be performed, because a class don't read files in the CSV format. Here, we almost just followed instructions, sometimes we did not really know how the results were achieved, being a truly complex process to understand. After concluding this step, the results were presented, but we also needed to make some "cleaning" on them, because the results presented were very specific and some information was redundant for our purpose. These tests were performed in the same conditions for both platforms, remembering that these *datasets* were *anonymized* using identical configurations, only this way was possible to prove, with a reasonable level of precision, that the results obtained were accurate.

The frameworks and source codes used to test the *anonymized dataset* are all made available by the University of *Aveiro* researchers, they also could be found on *GitHub* [115].

### 5.3. Results

In this section, we will present the conclusions reached on *re-identifying* the *anonymized datasets*. Moreover, these results were also very important before building the platform, because if the *dataset* produced by *ARX* did not at least match the level of protection achieved by the *Netflix dataset* against *re-identification* attacks, others options will need to be discussed. However, the results were quite satisfactory, even a little better than the *Netflix re-identification* attack results.

The literature found about this attack, helped a lot to understand the procedures and results produced by the algorithms and frameworks used, however, the process was not easy to follow, some terms and methods were very specific and hard to comprehend without studying these methods more deeply.

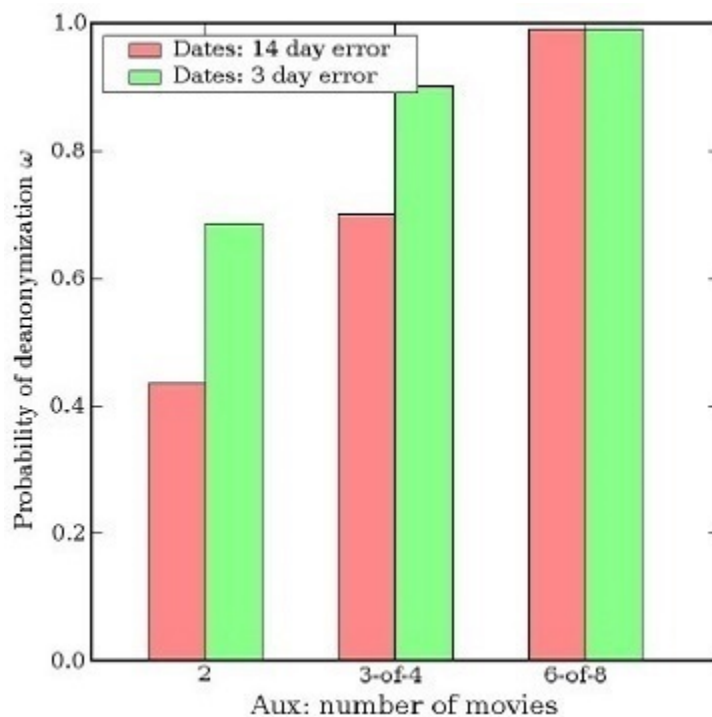


Figure 22: Narayanan et al. Re-identification Conclusions

Applying the *Scoreboard-RH* algorithm on the *Netflix dataset*, *Narayanan and Shmatikov* concluded that if an attacker known the exact rating of six to eight movies, the *re-identifying* risk were approximately 100% on both three and fourteen days error (figure 22). However, if the level of knowledge was only of two movies, for a three days error the risks decreased to about 78%, and for a fourteen days error was around 42% [17]. They also performed a similar study, in this one, they assumed that the attacker had zero knowledge in terms of dates.

Obviously, this reduced dramatically the *re-identifying* risks, knowing the exact rating of six to eight movies they estimated a 70% risk, and with a two movies knowledge the risks fall to around 5% [17]. This technique presented us a simple but interesting set of results, this because we could compare the levels of knowledge from this approach to the *ARX* models. When an attacker knows the rating dates he has some information on the *dataset*, so it can be compared to the prosecutor model, furthermore, the attacker's zero knowledge is similar to the journalist model.

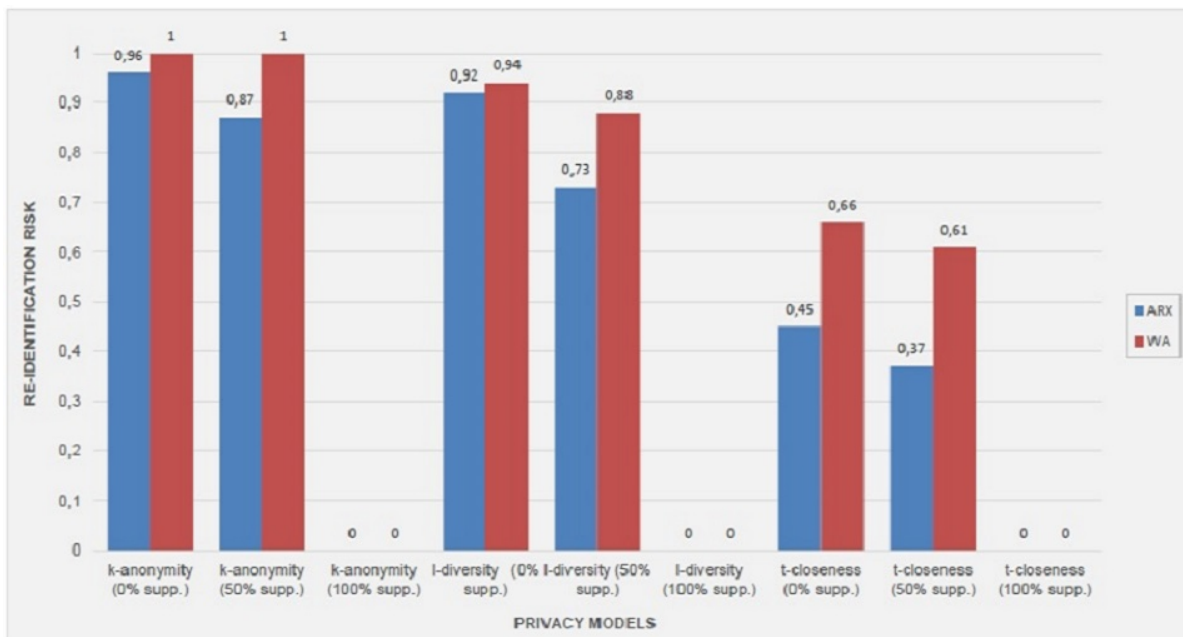


Figure 23: *Re-identification* Probability Results

The results obtained were in some way predicted, with the exception that the *Web Anonymizer* had slightly worst results than *ARX* (figure 23). We think this happened due to some *ARX* configurations not being implemented in the *Web Anonymizer*, the *dataset* is identical, and both runs the same API. The none implemented configurations help to obtain a more precise *anonymization process*, adding more protection to the *anonymized data*, so, we concluded that the differences observed were caused by those missing settings.

The *datasets* with 100% *supression* were far better than the other two, however, as seen before these *datasets* lose a lot of information, so, they more secure but less useful. The *supression* was applied to the "ClientID" attribute, being all suppressed is impossible to *re-identify* someone, that is why they have a 0% *re-identifying* risk.

Has said before in the "Procedure" section, we already were expecting poor results from the *k-anonymity* technique. With the platform created, this algorithm had a 100% probability of *re-identification*, but, don't forget that the attackers need to have some information about at least one client on the *dataset*. They need to know that the dates presented has a maximum of 14 days error, besides, they have to identify at least two movies or in the worst case eight of them.

The *ARX re-identification* risk results with some level of knowledge (prosecutor model) were: 81% on *k-anonymity*, 64% on *I-diversity* and 55% on *t-closeness*. On the other side, *Web Anonymizer* results were: 100% on *k-anonymity*, 73% on *I-diversity* and 65% on *t-closeness*. If that is not the case, i.e., if the user had zero knowledge the values obtained on both platforms were even better than the ones gathered by the attack presented before. The *ARX re-identification* risk results with zero knowledge (journalist model) were: 42% on *k-anonymity*, 18% on *I-diversity* and 3% on *t-closeness*. On the other hand, *Web Anonymizer* results were: 58% on *k-anonymity*, 23% on *I-diversity* and 11% on *t-closeness*.

We also noted that none of the *anonymized datasets* had a 0% of *re-identification* risk without using *supression*, i.e., every *dataset* tested were insecure and they could be somehow *re-identified* if the utility factor was safeguarded. However, these attacks need to have some prior knowledge of the *dataset* content, also, they require a lot of time, experience and powerful processing machines, in order to identify someone in a *dataset*.

Overall, both platforms demonstrated to be efficient, obtaining better results than the published *Netflix dataset anonymization* when submitted to the *re-identification* attack technique presented in this chapter. However, is not clear what were the techniques used to *de-identify* this *dataset*, *Netflix* claimed that a *sanitization* technique (*generalization*) was applied. But they didn't give more information about the *anonymization* process, so we cannot precise which algorithm tested was closer to the one used on *Netflix dataset*, if more details were given we could have presented even more precise results.

Resuming, the *Web Anonymizer* results were not exceptional, but not far from the *ARX* ones, in some way, is reassuring because this solution is one of the best available on the market. We think that with some improvements, this could be a good solution for *anonymizing datasets* over the *web*, however, we do not recommend using this application yet for really big *datasets*, with the consequence of crashing your *web* browser during the *anonymization* process.

# Conclusion and Future Work

The objectives of this dissertation were, initially, developing a *web* based platform that would allow the user to *anonymize datasets* with personal information. Besides that, an intuitive manner of presenting the results had to be found, also, the same solution was to be applied for showing to the user a risk analysis on the *anonymized dataset*. Obviously, we also needed to perform tests, in order to determine the robustness and resistance of the *anonymized* data produced by the platform against *re-identification* attacks.

Initially, a lot of solutions were tested, in order to select the best one to serve as a base for the *Web Anonymizer*, after, some *re-identification* tests were performed before creating the platform itself. This analysis was based on a successful *re-identification* attack performed over a released *Netflix dataset*, then executed on the *datasets anonymized* by the solution selected before. The intention was proving that the level of resistance to *re-identification* attacks, achieved by the *ARX* application, were better or at least similar.

After that, a development plan was created, designing an appropriate architecture for this *web* application, a conceptual model that presented the transitions between activities, a mockup modelling the interface visualization, also, a research for the best technologies available to create a *web* application.

Adapting the *ARX API* to our necessities was the next step, most of it was reused, but we also created some packages and classes. Most of these modifications had the intention to turn this API into a *web* version, so, all of that was wrapped up in a *RESTful API*. This way, all the communication between client and server were managed properly, to do that, we used AJAX as the request handler, also, this technology handles errors and success, presenting the results or messages to the user throughout an interactive interface.

## Conclusion

We believe that this project main objectives were accomplished, the *Web Anonymizer* works properly, producing *de-identified datasets* suitable to the user's demands. The platform allows the user to configure the *anonymization* process, defining the attribute type or the hierarchy levels, all this to produce the best *anonymized dataset*, according to the user's protection and utility standards. After the process being completed, the user can consult the results and a risk analysis, this cycle can be repeated as much as the user desire until he reaches the results wanted.

To guarantee the performance of the *Web Anonymizer*, and also, proving the resistance of the produced *datasets*, a series of tests were performed. Initially, we tested the execution times in comparison to the *ARX*, after we used a proven *re-identification* attack, determining the robustness of both platforms to this type of attacks.

In the first set of tests, *ARX* was much faster than the *Web Anonymizer*, actually, being in average approximately nine times faster. We believed that the principal cause of this is the delays observed over the Internet, remembering that we are talking about seconds of difference. Another possible cause is related to inappropriate coding, code optimizing could help reduce these times, also, having a powerful machine as server could also decrease the execution time. When tested for *re-identification* resistance, again, *ARX* had better results when compared to the *Web Anonymizer*. We performed the tests using two different scenarios (proposed by the attack authors), one assuming that the attacker had some knowledge of the *dataset* content, and another, where the attacker had zero knowledge. Besides, the three algorithms used were also tested in order to collect more data, this way, increasing the accuracy of the results obtained. In general, both platforms presented better outcomes than the one that were performed on the *Netflix dataset*, however, *ARX* was more resistant to *re-identification* attacks than the *Web Anonymizer*, that was verified in every test executed.

Overall, *ARX* software has passed each test with distinction, in some cases, beating the *Web Anonymizer* by a large margin (execution time). We believe that there are two main reasons for these results, one for each type of test. In the execution time test, the fact of being an installed

software makes the difference, even when the applications were being built, when running tests locally (localhost), the divergence on execution time was already notable. The other reason, the one that makes the robustness so distinct between platforms, is related to the configurations not included in the application created. This is the only possible answer, because the API is the same, also the configurations available, as well as, the *dataset* tested. The fact that the *Web Anonymizer* works over the Internet as nothing to do with it, so this is the only characteristic found that can affect the resistance of a *anonymized dataset* produced by this platform to *re-identification* attacks.

The application has some minor issues, these are presented in the topic below, together with some possible solutions, nonetheless, these problems have little impact on the *anonymization* process, they are more linked to design, result presentation and project structuring. Nonetheless, we believe that with the changes proposed below, and some others that could arise, the *Web Anonymizer* can reach performances near to the *ARX* solution, of course, they are some limitations that can be mitigated, but they cannot be eliminated at all.

With this platform, is possible to use the *anonymized datasets* produced for profit, that is already happening, but much of them don't respect the person's privacy. Besides, there are laws to obey and Europe will see them get more severe in 2018, so the *Web Anonymizer* could be a useful tool to transform a *dataset* into a law abiding one.

Summing up, we conclude that a significant contribution in terms of literature and application of technologies was done, resulting in a step forward regarding *datasets anonymization* with personal information over the Internet. This should become a common practice between all the entities that stores and processes this type of information, and most importantly, always respecting the person's right to privacy.



## Future Work

Based on the previous test results, and on the insights gained during this work, a collection of possible lines of work has been identified, we listed a set of topics that could be rectified or improved on the *Web Anonymizer*. These suggestions point to different aspects of the application, such as, improving interface design, enhancing file storage, complementing result presentation, improving hierarchies definition, upgrade configurations, code reviewing or including a login mechanism.

Below is presented the list of potential tasks:

- On the "Results" tab, the tables loaded have an independent scroll, to see the same records in both tables simultaneously will require that the scrolls function as one, allowing the user to see the differences between them at the same time.
- In the same view, it would be interesting to see a line numeration in the original and *anonymized datasets*, in order to give a better guidance to the user when his analysing the data. For that is recommended to create a database connection, configure it to properly, obtaining an indexed database that easily will present a column with sequential numeration.
- In this version, the storage of files is done locally on the server, we designed it this way due to the *ARX API*. This software only works with *datasets* on files, this creates a lot of limitations for this platform, as mentioned before, in the future should be implemented a database to store all the *datasets*.
- When the resulting *dataset* is presented, in the "Results" tab, is done in a simple way, showing the *anonymized* records in the same manner as the original *dataset*. We thought that in the future, some kind of highlighting or colouring should be done, i.e, the rows that had been modified should be presented in a different way, indicating the user what was *anonymized*. We thought enhancing this idea into a colouring system, e.g., the records that present severe *re-identification* risks should have a red background, the ones that introduce high risks should have an orange background, and so on for the other risk levels.

- Initially, one objective was presenting the risk analysis in a graphical manner, but this required to implement a Graphical User Interface (GUI), so we dropped that due to the complexity and time limitations. Presenting results in a graphical way is always attractive and easier to interpret, so, seeing that in a future version will increase the application quality.
- The hierarchy definition should also be improved, with a better representation of the levels. Maybe an automatic system for setting the top and bottom values, this will help a lot the user defining these intervals. The platform needs to accept non-numeric values, furthermore, a big step will be discriminate and use different methods for categorical and ordinal data. Also, a system to download the hierarchies configuration will prevent the repetitive task of defining these levels.
- All the written code, especially the *JavaScript*, should be revised and optimized by someone with more experience in creating *web* applications. This could correct some minor issues, and also, improve the execution time affected by the server delay.
- In a future revision, should be implemented all the other configuration options found in *ARX* software, this will turn the platform more complete, making the *anonymization* results more precise and robust.
- An interesting approach will be creating a login mechanism, this way, the user will be able to save his *datasets* (originals and *anonymized*), and he could keep a copy on the server, as well as, on his personal computer.

Another aspect taken into account was scalability, thinking in future upgrades, or even to be applied in other fields rather than the healthcare environment. When *ARX API* was adapted into the *Web Anonymizer*, we also have taken into consideration that in the future some features could be added, so we decided to keep the entire API without changes even if those classes were not used. Besides, all that was added to the original API (classes and methods) are well documented in this dissertation, so, consulting it will help to understand the work done, helping the regular user or for future developments.

The platform layout was also created thinking in future modifications, a lot space was left for including new elements. Furthermore, *HTML5* allows very easily the incorporation of new elements, with a more standard codification, increasing the semantic value and reducing the number of tags, simplifying the structure of HTML code. This way, we believe that expanding the *Web Anonymizer*, or even, a complete reconstruction, will not be a difficult task for a person with basic knowledge on *web* application developing.

# References

- [1] Elmagarmid A. and Sheth A. (2008) *Privacy-Preserving Data Mining: Models and Algorithms*. Springer Science+Business Media, LLC, New York, USA.
- [2] El Emam K. (2013) *Guide to the De-Identification of Personal Health Information*. CRC Press Taylor & Francis Group, Boca Raton, Florida, USA.
- [3] Raghunathan B. (2008) *The Complete Book of Data Anonymization - From Planning to Implementation*. CRC Press Taylor & Francis Group, Boca Raton, Florida, USA.
- [4] D'Acquisto G., Domingo-Ferrer J., Kikiras P., Torra V., Montjoye Y. and Bourka A. (2015) *Privacy by design in big data*. European Union Agency for Network and Information Security (ENISA), Science and Technology Park of Crete (ITE), Heraklion, Greece.
- [5] Neamatullah I., Douglass M., Lehman L., Reisner A., Villarroel M., Long W., et al. (2008) *Automated de-identification of free-text medical records*. BioMed Central, London, United Kingdom.
- [6] Margolis R., Derr L., Dunn M., Huerta M., Larkin J., Sheehan J., et al. (2014) *The National Institutes of Health's Big Data to Knowledge (BD2K) initiative: capitalizing on biomedical big data*. National Institute of Diabetes and Digestive and Kidney Diseases, NIH, Bethesda, Maryland, USA.
- [7] The IT Infrastructure Technical Committee. (2014) *IHE IT Infrastructure Handbook*. Integrating the Healthcare Enterprise, Oak Brook, Illinois USA.
- [8] Cavoukian A. and El Emam K. (2014) *De-identification Protocols: Essential for Protecting Privacy*. Information and Privacy Commissioner, Ontario, Canada.
- [9] Polonetsky J., Tene O. and Jerome J. (2014) *Benefit-Risk Analysis for Big Data Projects*. Future of Privacy Forum, Washington DC , USA.

- [10] Sariyar M. and Schlünder I. (2016) *Reconsidering Anonymization-Related Concepts and the Term "Identification" Against the Backdrop of the European Legal Framework*. Biopreservation and Biobanking Volume 14, Number 5, 2016, Mary Ann Liebert, Inc..
- [11] Duncan G., Elliot M. and Salazar-Gonzalez J. (2016) *Statical Confidentiality: Principles and Practice*. Springer Science+Business Media, LLC, New York, USA.
- [12] Scaiano M., Middleton G., Arbuckle L., Kolhatkar V., Peyton L., Dowling M., et al. (2016) *A unified framework for evaluating the risk of re-identification of text de-identification tools*. Elsevier Inc., Journal of Biomedical Informatics 63, 174–183, Amsterdam, Netherlands.
- [13] Sweeney L. (2002) *k-Anonymity: A Model for Protecting Privacy*. School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.
- [14] Sweeney L. (2002) *Achieving k-Anonymity Privacy Protection using Generalization and Suppression*. School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.
- [15] Ohm P. (2010) *Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization*. University of Colorado Law School (UCLA) Law Review 1701.
- [16] Samarati P. and Sweeney L. (1998) *Generalizing Data to Provide Anonymity when Disclosing Information*. Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA.
- [17] Narayanan A. and Shmatikov V. (2008) *Robust De-anonymization of Large Sparse Datasets*. Proceedings of the 2008 IEEE Symposium on Security and Privacy, Pages 111-125, IEEE Computer Society Washington, Washington DC, USA.
- [18] Google for Work. (1998) *Google for Work Security and Compliance Whitepaper*. Google Inc., Mountain View, California, EUA.
- [19] Independent European advisory body on data protection and privacy. (2007) *European Union Article 29 Data Protection Working Party, Opinion 4/2007 on the Concept of Personal Data*.

European Commission, Directorate General Justice, Freedom and Security, Brussels, Belgium.

- [20] Toubiana V. and Nissenbaum H. (2011) *Analysis of Google Logs Retention Policies*. Journal of Privacy and Confidentiality, Number 1, 3–26, Carnegie Mellon University, Pittsburgh, Pennsylvania, EUA.
- [21] European Parliament, Council of Europe. (2016) *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)*. Official Journal of the European Union, L-119, pp. 1-88.
- [22] El Emam K., Jonker E., Arbuckle L. and Malin B. (2011) *A Systematic Review of Re-Identification Attacks on Health Data*. Johns Hopkins Bloomberg School of Public Health, Baltimore, Maryland, USA.
- [23] El Emam K., Rodgers S. and Malin B. (2015) *Anonymising and sharing individual patient data*. The BMJ, London, UK.
- [24] Neubauer T. and Heurix J. (2011) *A methodology for the pseudonymization of medical data*. Elsevier Inc., Journal of Biomedical Informatics 80, 190–204, Amsterdam, Netherlands.
- [25] Riedl B., Neubauer T., Goluch G., Boehm O., Reinauer G. and Krumboeck A. (2010) *A secure architecture for the pseudonymization of medical data*. Second International Conference on Availability, Reliability and Security (ARES'07), IEEE.
- [26] Independent European advisory body on data protection and privacy. (2014) *European Union Article 29 Data Protection Working Party, Opinion 05/2014 on Anonymization Techniques*. European Commission, Directorate General Justice, Freedom and Security, Brussels, Belgium.
- [27] Murphy S. (2013) *ISO/TS 25237:2008 Overview*. SLMS Health Informatics, London, UK.

- [28] European Parliament, Council of Europe. (2001) *REGULATION (EC) No 45/2001 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 18 December 2000 on the protection of individuals with regard to the processing of personal data by the Community institutions and bodies and on the free movement of such data*. Official Journal of the European Communities, L-8, pp. 1-22.
- [29] El Emam K., Dankar F., Neisa A. and Jonker E. (2013) *Evaluating the risk of patient re-identification from adverse drug event reports*. BMC Medical Informatics and Decision Making 2013, 13:114.
- [30] Albrecht J. (2013) *Report on the proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation)*. Committee on Civil Liberties, Justice and Home Affairs.
- [31] European Data Protection Supervisor. (2005) *Position paper on the role of Data Protection Officers in ensuring effective compliance with Regulation (EC) 45/2001*. European Data Protection Supervisor, Brussels, Belgium.
- [32] The Centre for Information Policy Leadership. (2013) *The Role and Function of a Data Protection Officer in Practice and in the European Commission's Proposed General Data Protection Regulation*. The Centre for Information Policy Leadership, Hunton & Williams LLP, Brussels, Belgium.
- [33] Wubben M. (2013) *Factsheet Data Protection Officer*. Considerati, IT Law & Policy Specialists, Amsterdam, Netherlands.
- [34] Thanvi S. (2001) *Law of Torts*. University of Jodhpur, Rajasthan, India.
- [35] DeWolf D. (2009) *The Law of Torts: Cases and Materials*. Lupus Publications, Ltd., Lansing, Michigan, USA.

- [36] Jones W. (1985) *The Constitution of the People's Republic of China*. Washington University Law Review, Volume 63, Issue 4, Washington DC, USA.
- [37] de Hert P. and Papakonstantinou V. (2015) *The Data Protection Regime in China*. European Union, Brussels, Belgium.
- [38] National People's Congress Standing Committee. (2015) *National People's Congress Standing Committee Decision concerning Strengthening Network Information Protection*.
- [39] Xiao Dong M. (2014) *Data protection in China: Overview*. PLC, Thomson Reuters, London, UK.
- [40] Sotto L. and Segalis B. (2008) *Russia Launches a Data Protection Website*. Client Alert, August 200, Hunton & Williams LLP, New York, USA.
- [41] The State Duma. (2006) *Russian Federation Federal Law No. 152-FZ on Personal Data*. The Federation Council, Kremlin, Moscow, Russia.
- [42] The State Duma. (2006) *Russian Federation Federal Law No. 149-FZ on Information, Informational Technologies and the Protection of Information*. The Federation Council, Kremlin, Moscow, Russia.
- [43] Medvedev S. (2016) *Data protection in Russian: Overview*. PLC, Thomson Reuters, London, UK.
- [44] The State Duma. (2011) *Russian Federation Federal Law No. 323-FZ on the Fundamentals of Protection of the Health of Citizens in the Russian Federation*. The Federation Council, Kremlin, Moscow, Russia.
- [45] Burger E., Field M. and Twigg J. (1998) *From Assurance to Insurance in Russian Health Care: The Problematic Transition*. American Journal of Public Health, Vol. 88, No. 5, American Public Health Association, Washington DC, USA.



- [46] International Telecommunication Union (ITU), United Nations Population Division, Internet & Mobile Association of India (IAMAI), World Bank. (2016) *Internet Users by Country (2016)*. Internet Live Stats.
- [47] Lucente K. and Clark J. (2012) *Data Protection Laws of the World*. DLA Piper, London, UK.
- [48] Mathias S. and Kazia N. (2015) *Data protection in India: Overview*. PLC, Thomson Reuters, London, UK.
- [49] United Nations. (1948) *Universal Declaration of Human Rights*. United Nations, Manhattan, New York, USA.
- [50] Council of Europe. (1981) *Convention for the Protection of Individuals with regard to Automatic Processing of Personal Data*. Council of Europe, Strasbourg, France.
- [51] European Parliament, Council of Europe. (1995) *DIRECTIVE 95/46/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data*. Official Journal of the European Communities, L-281, pp. 31-50.
- [52] Boillat P. and Kjaerum M. (2014) *Handbook on European data protection law*. European Union Agency for Fundamental Rights, Vienna, Austria.
- [53] Council of Europe. (2007) *Working Document on the processing of personal data relating to health in electronic health records*. European Commission, Directorate-General Justice, Freedom and Security, Brussels, Belgium.
- [54] European Parliament, Council of Europe. (1981) *DIRECTIVE 2011/24/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 9 March 2011 on the application of patients' rights in cross-border healthcare*. Official Journal of the European Union, L-88, pp. 45-65.
- [55] Commission to the European Parliament, the Council of Europe, the European Economic and Social Committee and the Committee of the Regions. (2012) *eHealth Action Plan 2012-2020 - Innovative healthcare for the 21st century*. European Commission, Brussels, Belgium.

- [56] Appari A., Johnson M. and Anthony D. (2009) *HIPAA Compliance: An Institutional Theory Perspective*. Americas Conference on Information Systems (AMCIS), 2009, San Francisco, California, USA.
- [57] Garfinkel SL. (2016) *NIST Special Publication 800-188 - De-Identifying Government Datasets*. National Institute of Standards and Technology, US Department of Commerce, Gaithersburg, Maryland, USA.
- [58] Garfinkel S. (2015) *NISTIR 8053 - De-Identification of Personal Information*. National Institute of Standards and Technology, US Department of Commerce, Gaithersburg, Maryland, USA.
- [59] Federal Trade Commission. (2006) *Federal Trade Commission Act*. Federal Trade Commission, Washington DC, USA.
- [60] Senate and House of Representatives of the United States of America. (1996) *Health Insurance Portability and Accountability Act of 1996*. US Congress, Washington DC, USA.
- [61] Jolly I. (2016) *Data protection in the United States: overview*. PLC, Thomson Reuters, London, UK.
- [62] Malin B. (2012) *Guidance Regarding Methods for De-identification of Protected Health Information in Accordance with the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule*. Office for Civil Rights (OCR), U.S. Department of Health & Human Services, Washington DC, USA.
- [63] Lam P., Mitchell J. and Sundaram S. (2016) *A Formalization of HIPAA for a Medical Messaging System*. Stanford University, California, USA.
- [64] Department of Health and Human Services. (2016) *Breach Notification for Unsecured Protected Health Information; Interim Final Rule*. Federal Register, Vol. 74, No. 162, Washington DC, USA.
- [65] Department of Health and Human Services. (2013) *Modifications to the HIPAA Privacy, Security, Enforcement, and Breach Notification Rules Under the Health Information Technology*

*for Economic and Clinical Health Act and the Genetic Information Nondiscrimination Act; Other Modifications to the HIPAA Rules.* Federal Register, Vol. 78, No. 17, Washington DC, USA.

- [66] Rosenbaum S. (2013) *Law and the Public's Health*. George Washington University School of Public Health and Health Services, Department of Health Policy, Washington, DC, USA.
- [67] Senate and House of Representatives of the United States of America. (2015) *Judicial Redress Act of 2015*. US Congress, Washington DC, USA.
- [68] Portuguese Parliament. (1998) *Law 67/98 Act on the Protection of Personal Data*. Republic Diary, Series I-A, 247/1998, Portugal.
- [69] Cocco M. and Ornelas I. (2014) *Data protection in India: Overview*. PLC, Thomson Reuters, London, UK.
- [70] Portuguese Parliament. (1994) *Law 2/94 Establishes the Control and Verification Mechanisms for the Schengen Information System*. Republic Diary, Series I-A, 2/1994, Portugal.
- [71] Portuguese Parliament. (2005) *Portuguese Republic Constitution: Seventh Constitutional Revision - 2005*. Republic Diary, Series I-A, 155/2005, Portugal.
- [72] Portuguese Parliament. (2015) *Law 12/2015 Personal Genetic Information and Health Information*. Republic Diary, Series I-A, 18/2005, Portugal.
- [73] Thompson C. (2008) *If You Liked This, You're Sure to Love That*. The New York Times Magazine, 21 Nov 2008, New York, USA.
- [74] Netflix. (2006) *The Netflix Prize Rules*. Netflix, Inc., Los Gatos, California, USA.
- [75] Koren Y., Bell R. and Volinsky C. (2009) *Matrix Factorization Techniques for Recommender Systems*. IEEE Computer Society, 0018-9162/09, Washington DC, USA.
- [76] Dwork C. (2009) *An Ad Omnia Approach to Defining and Achieving Private Data Analysis*. Springer, Heidelberg, Berlin, Germany.

- [77] Christenson S. (2011) *Data breach exposes 4.9 million TRICARE patients*. mySA, Oct. 29, Hearst Newspapers, San Antonio, California, USA.
- [78] Conn J. (2011) *TRICARE reports data breach affecting 4.9 million patients*. Modern Healthcare, Oct. 29, Crain Communications, Inc., Detroit, Michigan, USA.
- [79] Aitoro J. (2014) *Computer tape theft exposing personal info of 5 million people? "Hardly a black-ops caper" judge says*. Washington Business Journal, May 14, Arlington, Vancouver, USA.
- [80] Miles D. (2011) *TRICARE investigates beneficiary data breach*. American Forces Press Service, Oct. 11, US Department of Defense, The Pentagon, Arlington County, Virginia, USA.
- [81] Versel N. (2011) *Military Health Plan Data Breach Threatens 4.9 Million*. Dark Reading, April 10, UBM, San Francisco, California, USA.
- [82] Anderson H. (2011) *TRICARE Breach Affects 4.9 Million*. Info Risk Today, Sep. 29, Princeton, New Jersey, USA.
- [83] McGee M. (2014) *Most Claims in TRICARE Breach Dismissed*. Data Breach Today, May 12, Princeton, New Jersey, USA.
- [84] Prasser F., Kohlmayer F., Lautenschläger R. and Kuhn K. (2014) *ARX - A Comprehensive Tool for Anonymizing Biomedical Data*. Proceedings of the AMIA 2014 Annual Symposium, Washington DC, USA.
- [85] Kohlmayer F., Prasser F. and Kuhn K. (2015) *The cost of quality: Implementing generalization and suppression for anonymizing biomedical data with minimal information loss*. Elsevier Inc., Journal of Biomedical Informatics 58, 37–48, Amsterdam, Netherlands.
- [86] Xiao X., Wang G. and Gehrke J. (2011) *CAT: The Cornell Anonymization Toolkit*. Department of Computer Science, Cornell University, Ithaca, New York, USA.

- [87] Xiao X., Wang G. and Gehrke J. (2009) *Interactive Anonymization of Sensitive Data*. Department of Computer Science, Cornell University, Ithaca, New York, USA.
- [88] Kantarcioglu M., Inan A. and Kuzu M. (2010) *Anonymization Toolbox*. Erik Jonsson School of Engineering & Computer Science, The University of Texas at Dallas, Texas, USA.
- [89] Templ M., Kowarik A. and Meindl B. (2015) *Statistical Disclosure Control for Micro-Data Using the R Package sdcMicro*. Journal of Statistical Software, Volume 67, Issue 4, Foundation for Open Access Statistics.
- [90] Meindl B., Templ M. and Kowarik A. (2013) *Guidelines for the Anonymization of Microdata Using R-package sdcMicro*. Data-Analysis OG, Viena , Austria.
- [91] Privacy Analytics. (2009) *Privacy Analytics Eclipse*. Privacy Analytics, Inc. Ottawa, Ontario, Canada.
- [92] Dai C., Ghinita G., Bertino E., Byun J. and Li N. (2009) *TIAMAT: a Tool for Interactive Analysis of Microdata Anonymization Techniques*. Very Large Data Base (VLDB), Lyon, France.
- [93] Poulis G., Gkoulalas-Divanis A., Loukides G., Skiadopoulos S. and Tryfonopoulos C. (2014) *SECRETA: A System for Evaluating and Comparing Relational and Transaction Anonymization algorithms*. EDBT: 17th International Conference on Extending Database Technology, 2014, Athens, Greece.
- [94] Aggarwal C. (2005) *On k-Anonymity and the Curse of Dimensionality*. IBM T. J. Watson Research Center, Yorktown Heights, New York, USA.
- [95] Dalenius T. (1986) *Finding a Needle In a Haystack or Identifying Anonymous Census Records*. Journal of Official Statistics, Vol.2, No.3, 1986. pp. 329–336, Sweden.
- [96] Sweeney L. (2002) *k-anonymity: A Model for Protecting Privacy*. IEEE Security and Privacy 1998, Washington DC, USA.

- [97] Sweeney L. (2002) *Achieving k-anonymity Privacy Protection using Generalization and Suppression*. IEEE Security and Privacy 1998, Washington DC, USA.
- [98] Nergiz M., Atzori M. and Clifton C. (2007) *Hiding the Presence of Individuals from Shared Databases*. ACM Special Interest Group on Management of Data (SIGMOD), 2007, Beijing, China.
- [99] Machanavajjhala A., Gehrke J. and Kifer D. (2007) *l-Diversity: Privacy Beyond k-Anonymity*. ACM Transactions on Knowledge Discovery from Data, Vol. 1, Issue 1, March 2007, New York, USA.
- [100] Brickell J. and Shmatikov V. (2008) *The Cost of Privacy: Destruction of Data-Mining Utility in Anonymized Data Publishing*. Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD), 2008, Las Vegas, Nevada, USA.
- [101] Li N., Li T. and Venkatasubramanian S. (2007) *Closeness: A New Privacy Measure for Data Publishing*. IEEE Transactions on Knowledge and Data Engineering, Vol. 22, Issue 7, July 2010, Washington DC, USA.
- [102] Zhang Q., Koudas N., Srivastava D. and Yu T. (2007) *Aggregate Query Answering on Anonymized Tables*. IEEE 23rd International Conference on Data Engineering, 2007, Istanbul, Turkey.
- [103] Rebollo-Monedero D., Forné J. and Domingo-Ferrer J. (2010) *From t-Closeness-Like Privacy to Postrandomization via Information Theory*. IEEE Transactions on Knowledge and Data Engineering, Vol. 22, Issue 11, November 2010, Washington DC, USA.
- [104] Prasser F., Eicher J., Bild R., Spengler H. and Kuhn K. (2017) *A Tool for Optimizing De-Identified Health Data for Use in Statistical Classification*. Technical University of Munich, University Hospital rechts der Isar, Institute of Medical Statistics and Epidemiology, Ismaninger Str. 22, 81675 Munich, Germany.

- [105] Fielding R. (2000) *Architectural Styles and the Design of Network-based Software Architectures*. University of California Irvine, California 92697, USA.
- [106] Richardson L. and Amundsen M. (2013) *RESTful Web APIs*. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, USA.
- [107] Allamaraju S. (2010) *RESTful Web Services - Cookbook*. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, USA.
- [108] Sweeney L. (2000) *Simple Demographics Often Identify People Uniquely*. Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.
- [109] Gonçalves J., Gomes D. and Aguiar L. (2015) *Privacy in Data Publishing for Tailored Recommendation Scenarios*. Transactions on Data Privacy, Vol. 8, Issue 1, 245–271, University of Skövde, Sweden.
- [110] Martin D., Kifer D., Machanavajjhala A., Gehrke J. and Halpern J. (2007) *Worst-Case Background Knowledge for Privacy-Preserving Data Publishing*. Cornell University, Ithaca, New York, USA.
- [111] Terrovitis M., Liagouris J., Mamoulis N., Gehrke J. and Skiadopoulos S. (2012) *Privacy Preservation by Disassociation*. Cornell University, Ithaca, New York, USA.
- [112] Zakerzadeh H., Aggrawal A. and Barker K. (2014) *Towards Breaking the Curse of Dimensionality for High-Dimensional Privacy: An Extended Version*. Cornell University, Ithaca, New York, USA.
- [113] Meyer B. (2006) *Netflix Recommender Framework*. <https://github.com/icefox/netflixrecommenderframework/tree/master/meyer> (Accessed: 14/04/2017)
- [114] Kadri S. (2005) *Kadri Framework*. <https://github.com/jmgoncalves/netflix-pseudonymizer/tree/master/kadri> (Accessed: 14/04/2017)

- [115] Gonçalves J. (2005) *Netflix Pseudonymizer*. <https://github.com/jmgoncalves/netflix-pseudonymizer> (Accessed: 14/04/2017)
- [116] Rogers Y., Sharp H. and Preece J. (2002) *Interaction Design: Beyond Human - Computer Interaction*. John Wiley & Sons, Inc., 605 Third Avenue, New York, NY 10158-0012, USA.
- [117] Norman D. (1988) *The Psychology of Everyday Things*. Basic Books, 387 Park Avenue South, New York, NY 10016-8810, USA.
- [118] Lamming M. and Newman W. (1995) *Interactive System Design*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.
- [119] Walls C. (2015) *Spring in Action*. Manning Publications Co., 20 Baldwin Road, Shelter Island, NY 11964, USA.
- [120] Walls C. (2016) *Spring Boot in Action*. Manning Publications Co., 20 Baldwin Road, Shelter Island, NY 11964, USA.
- [121] Kohlmayer F., Prasser F., Eckert C., Kemper A. and Klaus K. (2012) *Flash: Efficient, Stable and Optimal K-Anonymity*. 4th IEEE International Conference on Information Privacy, Security, Risk and Trust (PASSAT), Amsterdam, Netherlands.
- [122] Webb P., Syer D., Long J., Nicoll S., Winch R., Wilkinson A., et al. (2017) *Spring Boot Reference Guide*. Spring Boot, 2.0.0.BUILD-SNAPSHOT, 70 Mt. Hope Ave., P.O. Box 1609, Lewiston, ME 04240, USA.



# Appendix

## A API Documentation

The request methods available in the API implemented are documented in this Appendix. A small description of each method is presented, as well as, the type and parameters of the requests, the server answers and error messages, finally, request examples are specified in this document.

<b>URL</b>	/uploadFile
<b>Description</b>	Sends the file selected in the form to the server
<b>Method</b>	POST
<b>Parameters</b>	FormData
<b>Success Response</b>	Code: 200 (OK) Content: String
<b>Error Response</b>	Code: 400 (Bad Request) Content: String
<b>Code Sample</b>	<pre>\$.ajax({   url: "/uploadFile",   type: "POST",   data: new FormData(\$("#upload-file-form")[0]),   enctype: 'multipart/form-data',   processData: false,   contentType: false,   cache: false,   success: function () {     // Handle upload success     \$("#upload-file-message").text("File succesfully uploaded");   },   error: function () {     // Handle upload error     \$("#upload-file-message").text("File not uploaded (perhaps it's too much big)");   } });</pre>

<b>URL</b>	/anonym
<b>Description</b>	Send a JSON object with the configurations values to the server and receive an array with the <i>anonymization</i> process informations
<b>Method</b>	POST
<b>Parameters</b>	data = JSON.stringify({anonymType: anonymTypeV, anonymValue: anonymValueV, cValue: cValueV, tValue: tValueV, codiValue: codiValueV, atrib1Type: atrib1TypeV, atrib2Type: atrib2TypeV, atrib3Type: atrib3TypeV, atrib1Top: atrib1TopV, atrib1Bot: atrib1BotV, atrib1Int: atrib1IntV, atrib2Top: atrib2TopV, atrib2Bot: atrib2BotV, atrib2Int: atrib2IntV, atrib3Top: atrib3TopV, atrib3Bot: atrib3BotV, atrib3Int: atrib3IntV});
<b>Sucess Response</b>	Code: 200 (OK) Content: String
<b>Error Response</b>	Code: 400 (Bad Request), 500 (Internal Server Error) Content: String
<b>Code Sample</b>	<pre>\$.ajax({   url: "/anonym",   type: "POST",   data: data,   async: false,   dataType: 'json',   contentType: 'application/json',   cache: false,   success: function (data) {     \$("#timeSpent").text(data[0] + " segundos");     \$("#infoLossLow").text(data[1]);     \$("#infoLossHigh").text(data[2]);     \$("#suppRecords").text(data[3]);     \$("#attAnalysis1").text(data[4]);     \$("#attAnalysis2").text(data[5]);     \$("#attAnalysis3").text(data[6]);     \$("#attAnalysis4").text(data[7]);     \$("#attAnalysis5").text(data[8]);     \$("#attAnalysis6").text(data[9]);     \$("#attAnalysis7").text(data[10]);     \$("#prosecRisk").text(data[11]);     \$("#journRisk").text(data[12]);     \$("#markRisk").text(data[13]);   },   error: function () {     // Handle errors     var jsonResponse = JSON.parse(data.responseText);     alert(jsonResponse.message);   } });</pre>

<b>URL</b>	/downloadFile
<b>Description</b>	Make a request for the anonymized dataset
<b>Method</b>	GET
<b>Parameters</b>	filename: name
<b>Sucess Response</b>	Code: 200 (OK) Content: blob
<b>Error Response</b>	Code: 404 (Not Found) Code: 500 (Internal Server Error) Content: String
<b>Code Sample</b>	<pre>\$.ajax({   url: "/ downloadFile ",   type: "GET",   data: {filename: fileName},   success: function (data) {     var link = document.createElement('a');     link.href = window.URL.createObjectURL(data);     link.download = filename;     link.click();   },   error: function () {     // Handle download error     alert("Couldn't download the anonymized dataset");   } });</pre>

<b>URL</b>	/countRows
<b>Description</b>	Make a request for the number of rows that a file contains
<b>Method</b>	GET
<b>Parameters</b>	filename: name
<b>Sucess Response</b>	Code: 200 (OK) Content: String
<b>Error Response</b>	Code: 404 (Not Found) Code: 500 (Internal Server Error) Content: String
<b>Code Sample</b>	<pre>\$.ajax({   url: "/countRows",   type: "GET",   data: {filename: "original_datasets/test.csv"},   success: function (data) {     rowsTotal = data;     numPages = Math.floor(rowsTotal/rowsShown);   } }, error: function () {   // Handle dataset presentation error   \$("#upload-file-message").text("Error counting the dataset's rows"); } });</pre>

<b>URL</b>	/getRows
<b>Description</b>	Make a request for the 20 first lines of the uploaded file
<b>Method</b>	GET
<b>Parameters</b>	{filename: name, position: pos}
<b>Sucess Response</b>	Code: 200 (OK) Content: String
<b>Error Response</b>	Code: 500 (Internal Server Error) Content: String
<b>Code Sample</b>	<pre>\$.ajax({   url: "/getRows",   type: "GET",   data: {filename: fileName, position:pos},   success: function (data) {     var string = data + "";     for (var i=1; i&lt;20; i++) {       string = data[i] + "";       var cols = string.split(',');       \$('#uploadBody').append("&lt;tr&gt;" + "&lt;td&gt;" + (cols[0]) + "&lt;/td&gt;" + "&lt;td&gt;" + (cols[1]) +         "&lt;/td&gt;" + "&lt;td&gt;" + (cols[2]) + "&lt;/td&gt;" + "&lt;/tr&gt;");       \$('#originalBody').append("&lt;tr&gt;" + "&lt;td&gt;" + (cols[0]) + "&lt;/td&gt;" + "&lt;td&gt;" + (cols[1]) +         "&lt;/td&gt;" + "&lt;td&gt;" + (cols[2]) + "&lt;/td&gt;" + "&lt;/tr&gt;");     }   },   error: function () {     // Handle dataset presentation error     \$("#upload-file-message").text("Error presenting the dataset");   } });</pre>